

Representation Learning for Domain Adaptation and Cross-Modal Retrieval

**In the Context of Online Handwriting Recognition and
Visual Self-Localization**

Dissertation von Felix Ott

München 2023

Representation Learning for Domain Adaptation and Cross-Modal Retrieval

In the Context of Online Handwriting Recognition and Visual Self-Localization

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

eingereicht von
Felix Ott
20.04.2023

Erster Berichterstatter: Prof. Dr. rer. nat. Bernd Bischl
Zweiter Berichterstatter: Prof. Dr. rer. nat. Matthias Schubert
Dritter Berichterstatter: Prof. Dr.-Ing. habil. Andreas Maier
Tag der Disputation: 18.07.2023

Acknowledgments

Many wonderful people have accompanied and greatly supported me over the past four years to make this work possible. My deepest gratitude goes to ...

- ... first and foremost my supervisors Bernd Bischl and David Rügamer, for your unwavering support and guidance over all the years. I greatly appreciate the opportunity to have done my Ph.D. under your supervision. Their expertise and insight were invaluable in shaping my research and pushing me to new heights. David is a role model as a calm, credible, and empowering leader.*
- ... Matthias Schubert and Andreas Maier for acting as the second and third reviewer for this dissertation, and Fabian Scheipl and Thomas Nagler for your availability to be part of the examination panel at the disputation.*
- ... my mentor Christopher Mutschler. I would like to express my deepest gratitude for supporting my research direction at the Fraunhofer IIS.*
- ... my group leader Jochen Seitz for giving the freedom and time for research.*
- ... all members of the LMU SLDS chair, for engaging and productive discussions and the relaxed and supporting atmosphere.*
- ... my colleges from the Fraunhofer IIS. Particularly, I would like to thank Lucas Heublein, Nisha L. Raichur, Maximilian Stahlke, Tobias Feigl, Andreas Porada, Sebastian Kram, Jonathan Ott, Christoffer Löffler, and Georgios Kontes for your support and your brilliant comments. You have been there to support for data collections and fruitfull discussions. I also want to extend my appreciation to everyone who helped me to improve this thesis by meticulously reviewing portions of it.*
- ... all other collaborators who have shared their insights and ideas with me during many engaging discussions on various scientific topics.*
- ... to my parents, family, and friends. Words cannot express how grateful I am to my mother and father for all the sacrifices that you have made on my behalf.*

Summary

Most machine learning applications involve a domain shift between data on which a model has initially been trained and data from a similar but different domain to which the model is later applied on. Applications range from human computer interaction (e.g., humans with different characteristics for speech or handwriting recognition), computer vision (e.g., a change of weather conditions or objects in the environment for visual self-localization), and neural language processing (e.g., switching between different languages). Another related field is cross-modal retrieval, which aims to efficiently extract information from various modalities. In this field, the data can exhibit variations between each modality. Such variations in data between the modalities can negatively impact the performance of the model. To reduce the impact of domain shift, methods search for an optimal transformation from the source to the target domain or an optimal alignment of modalities to learn a domain-invariant representation that is not affected by domain differences.

The alignment of features of various data sources that are affected by domain shift requires representation learning techniques. These techniques are used to learn a meaningful representation that can be interpreted, or that includes latent features through the use of deep metric learning (DML). DML minimizes the distance between features by using the standard Euclidean loss, maximizes the similarity of features through cross correlation, or decreases the discrepancy of higher-order statistics like the maximum mean discrepancy. A similar but distinct field is pairwise learning and contrastive learning, which also employs DML. Contrastive learning not only aligns the features of data input pairs that have the same class label, but also increases the distance between pairs that have similar but different labels, thus enhancing the training process.

This research presents techniques for domain adaptation and cross-modal retrieval that specifically focus on the following two applications. (1) *Online handwriting recognition* involves representing written characters as multivariate time-series data from sensor-enhanced pens and aims to classify the written text. We recorded and evaluated various datasets for single character and sequence-to-sequence classification, and made them publicly available. We evaluated the domain shift that can occur between right- and left-handed writers, as well as between different writing styles, using uncertainty quantification techniques. Our approach utilizes higher-order statistics or optimal transport to adjust the features between right- and left-handed writers in order to minimize this domain shift. The best transformation is selected using DML techniques. Additionally, we assess the effectiveness of contrastive learning and DML for adapting the domain between writing on

tablet and on paper, as well as for cross-modal retrieval in offline and online handwriting recognition. (2) *Visual self-localization* aims to determine the absolute and relative position and orientation of a human or robot using only one monocular camera. We propose to enhance the task of predicting the absolute pose by incorporating an auxiliary task of predicting the relative pose using optical flow during the learning process and to pre-train on simulated data. In addition, we evaluate different fusion methods that utilize representation learning to combine information from visual and inertial sensors.

Keywords. representation learning · deep metric learning · domain adaptation · cross-modal retrieval · multi-modal fusion · time-series classification · online handwriting recognition · visual self-localization · pose regression

Zusammenfassung

Die meisten Anwendungen des maschinellen Lernens beinhalten die Herausforderung, dass eine Verschiebung der Verteilung zwischen Daten, auf denen ein Modell initial trainiert wurde, und Daten, aus einer ähnlichen aber unterschiedlichen Domäne, auf die das Modell später angewendet wird. Anwendungen reichen von Mensch-Maschine-Interaktion (z. B. Menschen mit unterschiedlichen Charakteristiken haben einen unterschiedlichen Einfluss auf die Sprach- oder Handschrifterkennung), Computer Vision (z. B. Veränderungen von Bedingungen oder Objekten in der Umgebung haben einen Einfluss auf die visuelle Eigenlokalisierung), und neuronale Sprachverarbeitung (z. B. der Wechsel zwischen verschiedenen Sprachen). Ein weiteres verwandtes Feld ist modalübergreifende Extraktion, sogenanntes “Cross-Modal Retrieval”, das darauf abzielt, Informationen aus verschiedenen Modalitäten effizient zu extrahieren. In diesem Bereich können Daten Unterschiede zwischen den verschiedenen Modalitäten aufweisen. Solche Veränderungen der Daten zwischen den Modalitäten können sich negativ auf die Leistung von Modellen auswirken. Um die Auswirkungen der Domänenverschiebung zu reduzieren, suchen Methoden nach einer optimalen Transformation der Merkmale vom Quell- zum Zielbereich oder eine optimale Ausrichtung der Modalitäten, um eine domäneninvariante Darstellung zu lernen, die nicht von Domänenverschiebung betroffen ist.

Um die Merkmale verschiedener Datenquellen, die von der Domänenverschiebung betroffen sind, aufeinander abzustimmen, sind Methoden notwendig, die eine optimale Darstellung lernen. Diese Darstellung sollte interpretierbar sein oder sollte latente Merkmale durch die Verwendung von tiefem metrischen Lernen, sogenanntem “Deep Metric Learning”, enthalten. Tiefes metrisches Lernen minimiert den Abstand zwischen Merkmalen mithilfe der üblich genutzten euklidischen Norm, maximiert die Ähnlichkeit von Merkmalen durch die Kreuzkorrelation, oder verringert die Diskrepanz von Statistiken höherer Ordnung, wie beispielsweise die maximale mittlere Diskrepanz. Ein ähnlicher, aber eigenständiger Bereich ist das kontrastive Lernen, das ebenfalls tiefes metrisches Lernen verwendet. Kontrastives Lernen gleicht nicht nur die Merkmale von Dateneingabepaaren an, die dieselbe Klasse vorweisen, sondern vergrößert auch den Abstand zwischen Dateneingabepaaren, die ähnliche, aber unterschiedliche Klassen vorweisen, und somit den Lernprozess verbessern.

Die Forschung, die in dieser Arbeit vorgestellt wird, präsentiert Methoden für die Domänenanpassung und die modalübergreifende Extraktion, die sich auf die im Folgenden beschriebenen zwei Anwendungen konzentrieren. (1) *Echtzeit-Handschrifterkennung* stellt geschriebene Zeichen als multivariate Zeitreihendaten dar, die mit Sensoren von speziell

entwickelten Stiften aufgenommen wurden, und hat das Ziel, den geschriebenen Text zu klassifizieren. Wir haben verschiedene Datensätze für die Klassifizierung von einzelnen Buchstaben und für die sequenzbasierte Klassifizierung aufgenommen, ausgewertet und öffentlich verfügbar gemacht. Wir haben die Domänenverschiebung, die zwischen Rechts- und Linkshänderdaten sowie zwischen verschiedenen Schreibstilen auftreten kann, unter Verwendung von Methoden zur Quantifizierung von Unsicherheiten ausgewertet. Unser Vorgehen verwendet Statistiken höherer Ordnung oder Methoden zur Bestimmung des optimalen Transports, um die Domänenverschiebung der Merkmale zwischen Rechts- und Linkshänderdaten zu minimieren. Die beste Transformation wird unter Verwendung von Methoden des tiefen metrischen Lernens ausgewählt. Zusätzlich bewerten wir die Wirksamkeit von kontrastivem Lernen und tiefem metrischen Lernen zur Anpassung der Domäne zwischen Sensordaten, die durch Schreiben auf einem Tablet und durch Schreiben auf einem Papier aufgenommen wurden, als auch für die modalübergreifende Extraktion in der zeitversetzten und Echtzeit-Handschrifterkennung. (2) Das Ziel der *visuellen Eigenlokalisierung* ist es, die absolute und relative Position und Orientierung eines Menschen, Roboters, oder eines sonstigen Objekts zu bestimmen, indem nur eine einzelne monokulare Kamera verwendet wird. Wir erweitern und verbessern den Prozess der absoluten Posenschätzung durch das Integrieren eines zusätzlichen Hilfsprozesses zum Vorhersagen der relativen Pose anhand des optischen Flusses während des Lernprozesses und durch vorheriges Training auf simulierten Daten. Darüber hinaus evaluieren wir verschiedene Fusionsmethoden, die Methoden des Representationslernens verwenden, um Informationen aus visuellen und inertialen Sensoren zu kombinieren.

Schlüsselwörter. Representationslernen · Tiefes metrisches Lernen · Domänenanpassung · Modalübergreifende Extraktion · Multimodale Fusion · Zeitserienklassifikation · Echtzeit-Handschrifterkennung · Visuelle Eigenlokalisierung · Posenregression

Contents

List of Figures	xiii
List of Tables	xvii
List of Abbreviations	xix
List of Notations	xxiii
I Introduction & Background	1
1 Introduction	3
1.1 Motivation & Overview	3
1.2 Applications	9
1.2.1 Online Handwriting Recognition	9
1.2.2 Visual Self-Localization	21
1.3 Deep Learning Foundations	27
1.3.1 Neural Networks	27
1.3.2 Temporal Networks	28
1.3.3 Time-Series Classification & Architectures	29
1.3.4 Feature Embeddings	31
1.4 Cross-Modal Retrieval & Multi-Modal Learning	32
1.4.1 Modalities	32
1.4.2 Methods & Fusion Points	33
1.4.3 Contrastive & Triplet Learning	35
1.5 Domain Adaptation	37
1.5.1 Definitions & Notations	38
1.5.2 Bound for Domain Adaptation	38
1.5.3 Categorization of Domain Adaptation Methods	39
1.5.4 Domain Adaptation with Optimal Transport	42
1.6 Representation Learning	45
1.6.1 Mean Squared Error & Frobenius Norm	48
1.6.2 Cosine Similarity & Pearson Correlation	49

1.6.3	Kullback-Leibler & Jensen-Shannon Divergence	51
1.6.4	Maximum Mean Discrepancy	51
1.6.5	Maximum (Mean and) Covariance Discrepancy	54
1.6.6	Correlation Alignment	54
1.6.7	Higher-Order Moment Matching	60
1.7	Dimensionality Reduction	62
2	Practical Applications of DA for Time-Series Classification	67
2.1	Benchmarking Toolbox & Contributions	67
2.2	Domain Adaptation Methods	69
2.3	Experimental Setup	72
2.3.1	Datasets	72
2.3.2	Architectures & Hyperparameters	78
2.4	Evaluation Results	80
II	Online Handwriting Recognition	107
3	The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning	109
4	Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach	133
5	Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift	153
6	Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens	167
7	Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift	201
8	Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition	221
9	Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition	265
III	Visual Self-Localization	279
10	ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization	281

11 Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression	297
12 Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments	329
Contributing Publications	363
Bibliography	367
Eidesstattliche Versicherung (Affidavit)	393
Glossary	395

List of Figures

1.1	Number of publications every year of topics covering this thesis.	5
1.2	Structural overview of a cross-modal and multi-task neural network model, as detailed in the corresponding section.	7
1.3	Exemplary datasets commonly used for offline HWR applications.	9
a	The MNIST dataset.	9
b	The IAM-OffDB dataset.	9
c	The George Washington dataset.	9
d	The Devanagari dataset.	9
1.4	Exemplary datasets commonly used for online trajectory-based and gesture-based HWR applications.	10
a	The PenDigits dataset.	10
b	The UJIpenchars dataset.	10
c	The VNOnDB dataset.	10
d	The ILGDB dataset.	10
e	The IAM-OnDB dataset.	10
f	The OnHW-wordsTraj dataset.	10
1.5	Sensor-enhanced pen with front and rear accelerometers, one gyroscope, one magnetometer, and one force sensor.	11
1.6	Exemplary sensor data of written symbols, words, and equations from the OnHW datasets.	14
a	OnHW-symbols. Label: '='.	14
b	OnHW-equations. Label: '28 · 3014282'.	14
c	OnHW-words500. Label: 'Quadratmeter'.	14
d	OnHW-wordsTraj. Label: 'Armweiler'.	14
1.7	Exemplary sensor signals (mean and standard deviation) for written words and equations.	15
a	Word samples.	15
b	Equation samples.	15
1.8	Exemplary sensor signals (mean and standard deviation) for words written on paper and tablet.	17
a	Word samples written on paper.	17
b	Equation samples written on tablet.	17
1.9	Visualization of the CTC loss function.	19

1.10	Visualization of the recording setup, challenges, and the point cloud in the Fraunhofer IIS L.I.N.K. test and application center.	23
a	Recording setup.	23
b	Motion blur and absorber wall.	23
c	Point cloud.	23
1.11	Overview of a multi-modal learning fusion architecture with early, intermediate, or late fusion.	34
1.12	General pipeline for DA methods that learn a domain-invariant representation between (labeled) source data and (unlabeled) target data.	37
a	Training pipeline.	37
b	Testing pipeline.	37
1.13	Overview of (unsupervised) DA categories and methods.	40
1.14	Differentiation between marginal, conditional, and joint domain alignment of source and target domain data samples.	41
a	Source and target domain data.	41
b	Marginal alignment: $P(\mathcal{U}_S) = P(\mathcal{U}_T)$	41
c	Conditional alignment: $P(\mathcal{Y}_S \mathcal{U}_S) = P(\mathcal{Y}_T \mathcal{U}_T)$	41
d	Joint alignment: $P(\mathcal{U}_S, \mathcal{Y}_S) = P(\mathcal{U}_T, \mathcal{Y}_T)$	41
1.15	Visualization of Sinkhorn that interpolates between OT and MMD distances.	44
a	OT distances.	44
b	MMD distances.	44
c	Sinkhorn distances.	44
1.16	Visualization of a representation encoded by a neural network to solve a classification task.	46
1.17	Structure of the singular value decomposition.	55
1.18	Visualization of the Bregman divergence.	58
1.19	Visualization of embeddings of OnHW time-series data using PCA and t-SNE.	65
a	Visualization of lowercase character embeddings.	65
b	Visualization of uppercase character embeddings.	65
c	Visualization of number and symbol embeddings.	65
d	Visualization of character embeddings recorded from right-handed writers, left-handed writers, and transformed left-handed writers.	65
1.20	Visualization of embeddings of generated time-series using PCA and t-SNE.	66
a	Sinusoidal embeddings without noise.	66
b	Sinusoidal embeddings with low noise.	66
c	Sinusoidal embeddings with intermediate noise.	66
d	Sinusoidal embeddings with higher noise.	66
e	Sinusoidal embeddings with highest noise.	66
2.1	Overview of dataset statistics of domain (legend) and class distributions.	76
a	HHAR.	76
b	UCI HAR.	76
c	WISDM.	76

d	SSC (EEG)	76
e	uWave.	76
f	Face detection.	76
g	GNSS.	76
h	Finger movements.	76
i	Gestures mid air.	76
2.2	Figure 2.1 continued.	77
a	Epilepsy.	77
b	PenDigits.	77
c	OnHW-symbols.	77
d	Split OnHW-equations.	77
e	OnHW-chars.	77
2.3	DA results for the GNSS-based dataset utilizing AdaTime for three encoder networks and for window lengths between 4 and 29.	90
a	CNN encoder.	90
b	ResNet18 encoder.	90
c	TCN encoder.	90
d	CNN encoder.	90
e	ResNet18 encoder.	90
f	TCN encoder.	90
2.4	DA results for the sinusoidal datasets with noise parameter b and for the backbones CNN, ResNet18, and TCN.	92
a	Methods MMDA and CDAN.	92
b	Methods CoDATS and DSAN.	92
c	Methods DANN and Deep CORAL.	92
d	Methods DIRT-T and HoMM _{$p=3$}	92
e	Methods DDC and AdvSKM.	92
2.5	Figure 2.4 continued.	93
a	Methods KL and JSD.	93
b	Methods Stein CORAL and Jeffreys CORAL.	93
c	Methods linear DAN and squared DAN.	93
d	Methods MSE and Sinkhorn.	93
e	Methods CS and PC.	93
2.6	Figure 2.5 continued.	94
a	Methods linear MMD and kMMD.	94
b	Methods linear MMCD and kMMCD.	94
2.7	Hyperparameter search for the Sinkhorn parameter γ	94
a	Human activity recognition datasets.	94
b	OnHW recognition datasets.	94
c	Sinusoidal datasets with noise parameter b	94
2.8	Figure 2.6 continued.	95
a	Methods Sinkhorn and Sinkhorn+HoMM _{$p=3$}	95
b	Methods Sinkhorn+linear MMD and Sinkhorn+kMMD.	95

c	Methods Sinkhorn+Deep CORAL, Sinkhorn+Jeffreys CORAL, and Sinkhorn+Stein CORAL.	95
2.9	Combination of Sinkhorn with KL for the weighting parameter ϵ	98
a	HHAR.	98
b	UCI HAR.	98
c	WISDM.	98
d	SSC.	98
e	uWave.	98
f	Finger.	98
g	Gestures mid air.	98
h	Epilepsy.	98
i	Face detection.	98
j	PenDigits.	98
k	OnHW-symbols.	98
l	Split OnHW-equations.	98
m	OnHW-chars.	98
n	Sinusoidal datasets.	98
2.10	DA results for different source to target domains for seven human activity and sleep recognition datasets exemplary for the CNN encoder and HoMM.	100
a	HHAR dataset.	100
b	UCI HAR dataset.	100
c	WISDM dataset.	100
d	SSC (EEG) dataset.	100
e	uWave dataset.	100
f	Finger dataset.	100
g	Gestures mid air dataset.	100
2.11	DA and risk results for the OnHW datasets and the CNN encoder for left-handed and right-handed writers.	101
a	Adaptation of right-handed writers to left-handed writers (R \rightarrow L).	101
b	Adaptation of left-handed writers to right-handed writers (L \rightarrow R).	101
2.12	Comparison of DA methods on the 10 HAR datasets w.r.t. the upper bound.	102
2.13	Comparison of DA methods on the three OnHW datasets w.r.t. the upper bound.	103
2.14	Hyperparameter search for the OnHW datasets (L \rightarrow R) for ten DA methods. Left: CNN encoder. Right: ResNet18 encoder.	104
a	OnHW-symbols.	104
b	Split OnHW-equations.	104
c	OnHW-chars (WD, combined).	104
2.15	Hyperparameter search for the OnHW datasets (L \rightarrow R) for the CNN encoder (left) and the ResNet18 encoder (right).	105
a	OnHW-symbols.	105
b	Split OnHW-equations.	105
c	OnHW-chars (combined).	105

List of Tables

1.1	Reference of the sections to the chapters of the corresponding, contributing paper.	8
1.2	Overview of the OnHW datasets (right-handed and left-handed writers) and state-of-the-art online handwriting datasets for writer-dependent (WD) and writer-independent (WI) tasks.	13
1.3	Summary of APR and RPR fusion techniques proposed in the visual-inertial self-localization papers.	25
2.1	Statistics about time-series datasets with domain shift.	72
2.2	Results for the source and target test datasets, evaluated on the model trained on target samples to show the discrepancy between source and target domains.	80
2.3	DA results in % (mean and standard deviation over all scenarios and five runs) for the HHAR dataset. Table 2.4 to Table 2.12 continues.	81
2.4	DA results: UCI HAR dataset.	82
2.5	DA results: WISDM dataset.	82
2.6	DA results: SSC (EEG) dataset.	83
2.7	DA results: uWave dataset.	83
2.8	DA results: Finger dataset.	84
2.9	DA results: Gestures mid air dataset.	84
2.10	DA results: Epilepsy dataset.	85
2.11	DA results: Face detection dataset.	85
2.12	DA results: PenDigits dataset.	86
2.13	DA results for the OnHW datasets. CNN as encoder network.	87
2.14	DA results for the OnHW datasets. ResNet18 as encoder network.	88
2.15	DA results for the OnHW datasets. TCN as encoder network.	89
2.16	DA results of the combination of Sinkhorn with alignment loss functions for the human activity recognition datasets.	96
2.17	Table 2.16 continued.	97
2.18	DA results of the combination of Sinkhorn with alignment loss functions for the OnHW recognition datasets.	97
2.19	Overview of training times (in s) exemplary on the sinusoidal dataset.	103

List of Abbreviations

APR	absolute pose regression
AU	aleatoric uncertainty
BiLSTM	bidirectional long short-term memory
CCC	concordance correlation coefficient
CE	cross-entropy
CER	character error rate
CMR	cross-modal retrieval
CNN	convolutional neural network
CORAL	correlation alignment
CRR	character recognition rate
CS	Cosine similarity
CTC	connectionist temporal classification
DA	domain adaptation
DL	deep learning
DML	deep metric learning
DNN	deep neural network
DTW	dynamic time warping
ECE	expected calibration error
ED	Edit distance
EEG	electroencephalography

EMD	Earth's Movers distance
EU	epistemic uncertainty
FC	fully connected
FCN	fully connected network
FL	focal loss
GAN	generative adversarial network
GCE	generalized cross-entropy
GMM	group moment matching
GNSS	global navigation and satellite system
GPS	global positioning system
GPU	graphics processing unit
GRU	gated recurrent unit
GTR	gated text recognizer
HAR	human activity recognition
HBS	boot hard
HoMM	higher-order moment matching
HWR	handwriting recognition
i.i.d.	independent and identically distributed
IMU	inertial measurement unit
JO	joint optimization
JSD	Jensen-Shannon divergence
NLP	natural language processing
kHoMM	kernalized higher-order moment matching
KL	Kullback-Leibler
kMMD	kernalized maximum mean discrepancy
L.I.N.K.	localization, identification, navigation, and communication

LM	language model
LSTM	long short-term memory
LSR	label smoothing
MAE	mean absolute error
MCD	maximum covariance discrepancy
MCU	multi-modal contextualization unit
ML	machine learning
MMCD	maximum mean and covariance discrepancy
MMD	maximum mean discrepancy
MMTM	multi-modal transfer module
MSE	mean squared error
MTL	multi-task learning
MTS	multivariate time-series
OCR	optical character recognition
OF	optical flow
OnHW	online handwriting
OT	optimal transport
PC	Pearson correlation
PCA	principal component analysis
PGO	pose graph optimization
RBF	radial basis function
ReLU	rectified linear unit
ResNet	residual network
RKHS	reproducing kernel Hilbert space
RMSE	root mean squared error
RNN	recurrent neural network

RPR	relative pose regression
RSM	random sampling matching
SBS	boot soft
SSF	selective sensor fusion
SCE	symmetric cross-entropy
SfM	structure from motion
SGD	stochastic gradient descent
SIFT	scale-invariant feature transform
SLAM	simultaneous localization and mapping
SSF	selective sensor fusion
SVD	singular value decomposition
SWAG	Stochastic Weight Averaging-Gaussian
seq2seq	sequence-to-sequence
TCN	temporal convolutional network
TL	transfer learning
TST	time-series transformer
t-SNE	t-distributed stochastic neighbor embedding
UQ	uncertainty quantification
VAE	variational autoencoder
ViPR	visual odometry-aided pose regression
VO	visual odometry
WD	writer-dependent
WER	word error rate
WI	writer-independent
6DoF	six degree-of-freedom

List of Notations

Notation	Description
General	
$f(\cdot)$	Function
\mathcal{L}	Loss function
$\ \cdot\ _2$	L_2 norm
$\ \cdot\ _F$	Frobenius norm
$M \in \mathbb{N}$	Number of modalities
$T \in \mathbb{N}$	Number of tasks (classification or regression)
\mathcal{X}	Set
\mathbf{X}	Matrix
\mathbf{x}	Vector
x	Scalar
$\sigma_{\mathbf{x}}$	Standard deviation of \mathbf{x}
$\bar{\mathbf{x}}$	Mean of \mathbf{x}
\mathbf{X}^T	Transpose of matrix \mathbf{X}
\mathbf{x}^T	Transpose of vector \mathbf{x}
$\text{Tr}(\mathbf{X})$	Trace of matrix \mathbf{X}
α, β, γ	Hyperparameter
b	Bias
$\mathbf{U}, \mathbf{W}, \mathbf{V}$	Weight matrices
\mathbf{w}	Weight vector
σ	Sigmoid activation
D	Dataset
θ	Network parameters
η	Learning rate
$p(\theta D)$	Posterior distribution
\otimes	Kronecker product
S_{++}^n	Space of $n \times n$ symmetric positive definite matrices
$(\cdot)^{\otimes p}$	p -level tensor power
bs	Batch size
$\mathcal{O}(\cdot)$	Time or space complexity

Notation	Description
Multivariate Time-Series (MTS) Classification (Ott et al., 2022a)	
$\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$	An MTS (ordered sequence) of $l \in \mathbb{N}$ streams
$\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l})$	A stream of an MTS with $i \in \{1, \dots, m\}$
$m \in \mathbb{N}$	Length of a time-series
$l \in \mathbb{N}$	Number of streams
$\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\}$	Set of n_U time-series with $\mathcal{U} \in \mathbb{R}^{n_U \times m \times l}$
$v \in \Omega$	Class label for a given MTS
Ω	Label space
$K = \Omega $	Number of class labels in the label space \mathcal{V}
$\mathbf{v} \in \Omega^L$	Sequence of L class labels
$\mathcal{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\} \in \Omega^{n \times L}$	Training labels of sequence samples
$\{\cdot\}$	Blank label
Domain Adaptation (DA) (Ott et al., 2022a)	
\mathcal{U}_S	Source domain dataset
\mathcal{U}_T	Target domain dataset
\mathcal{U}_{S_t}	Training subset of the source domain dataset
\mathcal{U}_{S_v}	Validation subset of the source domain dataset
\mathcal{U}_{T_t}	Training subset of the target domain dataset
\mathcal{U}_{T_v}	Validation subset of the target domain dataset
\mathbf{U}_S	MTS of the source domain dataset
\mathbf{U}_T	MTS of the target domain dataset
\mathbf{U}_{S_t}	Trainings subset of the source domain dataset
\mathbf{U}_{S_v}	Valdiation subset of the source domain dataset
\mathbf{U}_{T_t}	Trainings subset of the target domain dataset
\mathbf{U}_{T_v}	Valdiation subset of the target domain dataset
μ_S, μ_T	Mean of \mathcal{U}_S , respectively of \mathcal{U}_T
$\mathbf{C}_S, \mathbf{C}_T$	Covariance matrix of \mathcal{U}_S , respectively of \mathcal{U}_T
$f(\mathbf{U}_{S_t}) \in \mathbb{R}^{q_S \times w_S}$	Latent embedding of the training MTS of the source domain dataset
$f(\mathbf{U}_{T_t}) \in \mathbb{R}^{q_T \times w_T}$	Latent embedding of the training MTS of the target domain dataset
q_*, w_*	Size of the embedding $f(\mathbf{U}_{*+}) \in \mathbb{R}^{q_* \times w_*}$
\mathcal{X}	Feature space
$P(\mathcal{X})$	Marginal probability of \mathcal{X}
$P(\mathcal{X}, \mathcal{Y})$	Joint distribution of \mathcal{X} and \mathcal{Y}
$P(\mathcal{Y} \mathcal{X})$	Conditional distribution between \mathcal{X} and \mathcal{Y}
\mathcal{D}	Domain
$\mathcal{D}_S = \{\mathcal{X}_S^i, \mathcal{Y}_S^i\}_{i=1}^{\mathcal{N}_S}$	Source domain of \mathcal{N}_S labeled samples
$\mathcal{D}_T = \{\mathcal{X}_T^i, \mathcal{Y}_T^i\}_{i=1}^{\mathcal{N}_T}$	Target domain of \mathcal{N}_T labeled samples
$\mathbf{T} : \mathcal{D}_S \rightarrow \mathcal{D}_T$	Transformation of the source to the target domain
\mathcal{H}	Hypothesis space
$\mathcal{R}_{\mathcal{D}_S}$	Source domain error
$\mathcal{R}_{\mathcal{D}_T}$	Target domain error
\mathcal{H}	Hypothesis space, \mathcal{H} -divergence
$d_{\mathcal{H}\Delta\mathcal{H}}$	Discrepancy distance between \mathcal{D}_S and \mathcal{D}_T w.r.t. \mathcal{H}

Notation	Description
Triplet Learning	
$f(\mathbf{U}^a)$	Feature embedding of an anchor sample
$f(\mathbf{U}^p)$	Feature embedding of a positive sample
$f(\mathbf{U}^n)$	Feature embedding of a negative sample
$(f(\mathbf{U}_i^a), f(\mathbf{U}_i^p), f(\mathbf{U}_i^n)) \in \Theta$	Set of all training sample pairs
Sequence-to-Sequence Classification	
ED	Edit distance between two sequences
S_c	Character substitutions
I_c	Character insertions
D_c	Character deletions
N_c	Total number of characters in the set
S_w	Word substitutions
I_w	Word insertions
D_w	Word deletions
N_w	Total number of words in the set
$\mathbf{f} = (f_1, \dots, f_r)$	Sequence of length r
Optimal Transport (Ott et al., 2022a)	
$\mathbf{T} : \mathcal{D}_S \rightarrow \mathcal{D}_T$	Nonlinear map
$C(\mathbf{T})$	Transportation cost
\mathbf{T}_0	Optimal transportation problem
$\alpha \in \Theta$	General coupling
$\Theta \in P(\mathcal{D}_S, \mathcal{D}_T)$	Set of all probabilistic couplings
W_p	Wasserstein distance of order p
$c(\mathbf{U}_S, \mathbf{U}_T) : \mathcal{D}_S \times \mathcal{D}_T \rightarrow \mathbb{R}^+$	Cost function (symmetric, positive)
$d(\mathbf{U}_S, \mathbf{U}_T)^p$	Distance function of order p
$S_\epsilon(\mu_S, \mu_T)$	Sinkhorn divergence
ϵ	Regularization parameter

Notation	Description
Pose Regression (Ott et al., 2023b)	
APR_V	Absolute pose regression based on visual input
RPR_V	Relative pose regression based on visual input
RPR_I	Relative pose regression based on inertial input
$\mathbf{x} = [\mathbf{p}, \mathbf{q}]$	Absolute pose
$\Delta \mathbf{x} = [\Delta \mathbf{p}, \Delta \mathbf{q}]$	Relative pose
$\Delta \mathbf{x}^{tr} = [\Delta \mathbf{p}^{tr}, \Delta \mathbf{q}]$	Relative pose transformed
$\mathbf{p} \in \mathbb{R}^3$	Absolute 3D position in Euclidean space
$\mathbf{q} \in \mathbb{R}^4$	Absolute orientation as quaternion
$\Delta \mathbf{p} \in \mathbb{R}^3$	Relative position (translation)
$\Delta \mathbf{q} \in \mathbb{R}^4$	Relative orientation (rotation)
$\hat{\mathbf{p}} \in \mathbb{R}^3$	Ground truth absolute 3D position in Euclidean space
$\hat{\mathbf{q}} \in \mathbb{R}^4$	Ground truth absolute orientation as quaternion
$\Delta \hat{\mathbf{p}} \in \mathbb{R}^3$	Ground truth relative position (translation)
$\Delta \hat{\mathbf{q}} \in \mathbb{R}^4$	Ground truth relative orientation (rotation)
\mathbf{v}_{ij}	Relative pose between predicted poses \mathbf{x}_i and \mathbf{x}_j
$\mathbf{a}_I, \mathbf{a}_V$	Inertial and visual features

Part I

Introduction & Background

Chapter 1

Introduction

1.1 Motivation & Overview

It is common in numerous real-world scenarios to encounter a domain shift in the distribution of data between training and test sets. This leads to a decrease in the performance of machine learning (ML) models when applied to test data with a domain shift, as standard ML models assume that training and test datasets are *independent and identically distributed* (i.i.d.) (Sun et al., 2016). A domain shift, also known as distributional or covariate shift, pertains to a change in the data distribution between training data, which originates from a source domain, and related but dissimilar test data, which originates from a target domain (Ben-David et al., 2009). The subsequent four use cases exemplify common scenarios where domain shift arises: (1) In the field of human-computer interaction, models confront variable data sources due to individual differences in human characteristics (Tavenard et al., 2022; Shi et al., 2022; Ragab et al., 2023). This is particularly significant in the context of speech recognition, where distinctive speaking patterns among individuals influence the data distribution (Gu et al., 2022), or handwriting recognition, where variations in writing styles give rise to distinct writing patterns (Klaß et al., 2022). (2) In computer vision applications, such as those involving changes in weather conditions, variations in image quality occur consistently (Long et al., 2017; Tzeng et al., 2014). (3) Neural language processing applications experience domain shift as a result of differences in colloquial languages (Ben-David et al., 2009; Zhao et al., 2014). (4) The application of models in medical and diagnostic fields can pose a challenge when dealing with data from new diseases (Ozyurt et al., 2023). Transfer learning (Pan & Yang, 2009; Shao et al., 2014; Patel et al., 2015; Day & Khoshgoftaar, 2017) and domain adaptation (DA) (Tzeng et al., 2014; Long et al., 2017; Alipour & Tahmoresnezhad, 2021; Ma et al., 2022; Tavenard et al., 2022; Shi et al., 2022; Ragab et al., 2023; Ozyurt et al., 2023) compensates for this domain shift by transferring knowledge from a source to a target domain. Methods aim to identify an optimal transformation (Courty et al., 2016) that converts the (current) source domain data into the target domain by acquiring a domain-invariant representation that diminishes domain discrepancy.

A related field is cross-modal retrieval (CMR), which pertains to multi-modal learning that encompasses the topics of sensor and information fusion. CMR is concerned with learning across two or more modalities, where $M > 2$. A modality M refers to a distinct and independent channel of sensory input, such as image, video, audio, text, 3D data, graph information, or time-series data obtained from accelerometer, gyroscope, or magnetometer sensors. CMR has a broad range of applications, such as human activity recognition and handwriting recognition from both visual and inertial input (He et al., 2022a,b; Zhang et al., 2022; Shi et al., 2022; Singh & Chaturvedi, 2023), multimedia learning from visual input (Gu et al., 2022), and semantic embeddings from both visual and semantic input (Wang et al., 2022; Ohishi et al., 2022; Singh et al., 2022; Guzhov et al., 2022). Utilizing CMR to extract information from multiple modalities and adapt to the domain enables the utilization of information across all domains (Ranjan et al., 2015). The inconsistent representation form of various modalities, referred to as the *heterogeneity gap*, presents a significant challenge for CMR that has recently received considerable attention from the ML community (Huang et al., 2020). One category of techniques involves learning shared information between modalities using an information fusion module, such as MMTM (Joze et al., 2020) or soft fusion (Chen et al., 2019). Another approach is to learn an optimal embedding with representation learning (Huang et al., 2020; Brieger et al., 2022; Wang et al., 2022; Ohishi et al., 2022; Singh et al., 2022; Deldari et al., 2022a).

Representation learning can be advantageous for both DA and CMR. The goal of representation learning is to align features of different data sources to learn meaningful and interpretable representations. Deep metric learning (DML) or similarity learning is a subfield of representation learning that aims to learn a distance function between features of sub-domains. DML learns a weight matrix that can reduce the distance between similar points while increasing the distance between dissimilar points. Although the Euclidean distance or the Cosine similarity have been commonly used as metrics, recent works in DML have introduced alternative distances such as maximum mean discrepancy (MMD) (Borgwardt et al., 2006; Gretton et al., 2012), correlation alignment (CORAL) (Sun & Saenko, 2015), or higher-order moment matching (HoMM) (Chen et al., 2020) that consider higher-order statistics to align features. To align the features of corresponding data sources, DML usually takes into account pairs and minimizes the distance between the anchor and the positive samples. Contrastive and triplet learning methods define negative samples (i.e., samples with different class labels) and maximize the distance between the anchor and negative samples (Schroff et al., 2015). This approach is widely used for CMR.

To summarize the topics addressed in this thesis, we focus on representation learning and DML for DA and CMR, which have received significant attention from the ML research community. Figure 1.1 provides an overview of the number of publications on these topics between 2000 to 2022. In 2022, representation learning received significant attention with more than 390,000 publications. The number of publications on DML has been increasing strongly in recent years. While DA has received high attention, the term ‘cross-modal retrieval’ is mainly used within the ML community. Therefore, we focus on topics that have broad and increasing interest among researchers.

In this thesis, we introduce DML-based methods for addressing the topics DA and CMR

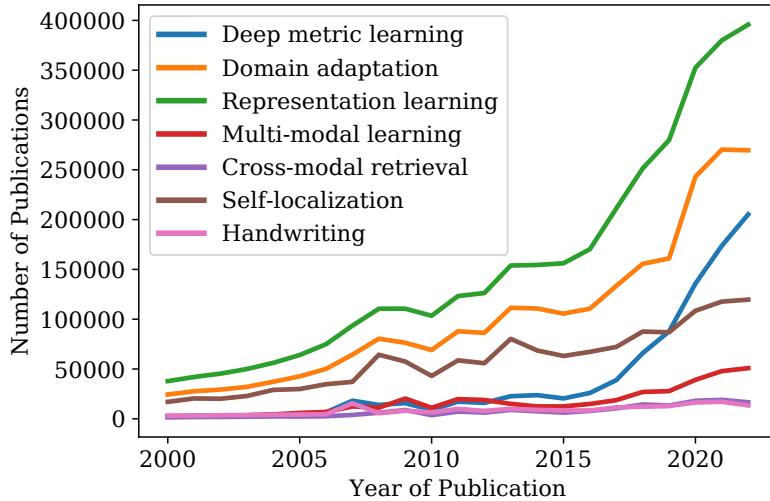


Figure 1.1: Number of publications every year of topics covering this thesis. Numbers from *Dimensions*: <https://app.dimensions.ai/discover/publication>.

in the context of two specific applications: Firstly, we focus on online handwriting (OnHW) recognition, which involves classifying written text that is represented as multivariate time-series (MTS) data collected from sensor-enhanced pens. The OnHW task is affected by a domain shift, which arises due to variations in writing styles among right-handed and left-handed writers, as well as variations in writing surfaces (paper versus tablet). DA techniques can enhance the performance of ML models. The second application is visual self-localization, which involves predicting the absolute and relative pose (position and orientation) of a moving object using a single monocular camera. We propose methods to enhance pose estimation utilizing multi-modal fusion techniques that incorporate information from both visual and inertial sensors. Additionally, we investigate the fusion of absolute and relative pose predictions to improve the accuracy of absolute pose estimation. The field of handwriting recognition (HWR) has maintained a consistent research interest of approximately 10,000 publications per year, while the number publications related to self-localization has been increasing, with approximately 100,000 publications per year, as shown in Figure 1.1.

We summarize our contributions from the published papers, which include proposing datasets and techniques for DA and CMR in the domain of OnHW recognition:

*C*₁ A novel sensor-enhanced pen is utilized in this research, enabling real-time and high-quality data collection. We record and publicly release diverse OnHW datasets comprising single letters (i.e., characters, numbers, and symbols) and letter sequences (i.e., words and mathematical equations). One of our contributions is the collection and publication of a dataset consisting of words written on both paper and tablet, which exhibit a domain shift between the two types of sensor data.

*C*₂ Our proposal entails the creation of a journal that focuses on the comparison of

techniques and datasets used in offline HWR and trajectory-based online HWR.

*C*₃ By computing statistics between trajectory-based and sensor-enhanced pen-based OnHW datasets, we are able to provide a technical overview of the recently published datasets.

*C*₄ Our contribution includes a dataset of single letters comprising characters, numbers, and symbols recorded on a tablet with a ground truth trajectory. We propose a joint classification and trajectory regression approach for single character recognition. By utilizing multi-task learning, we aim to enhance trajectory alignment, improve classification accuracy, and minimize trajectory prediction errors. We achieve this through the combination of cross-entropy loss with distance and similarity losses.

*C*₅ Our contribution is an evaluation benchmark for sequence-to-sequence and single character-based HWR, encompassing 28 methods that employ recurrent and temporal convolutional neural networks, as well as Transformer architectures. In addition, we assess the effectiveness of MTS augmentation techniques and cross-entropy variants.

*C*₆ Our study comprises a comprehensive evaluation of aleatoric (data) and epistemic (model) uncertainty quantification, utilizing two prominent Bayesian inference techniques: Stochastic Weight Averaging-Gaussian (SWAG) and Deep Ensembles. Our analysis enables the detection of out-of-distribution data and domain shifts that arise when combining handwriting samples from right-handed and left-handed writers.

*C*₇ In order to asses domain shift in time-series feature alignment, we propose a novel supervised DA method. This method searches for an optimal class-dependent transformation from the source to the target domain using a small number of samples. We compare DML similarity techniques (i.e., Cosine similarity, MMD, CORAL, and HoMM) to select the corresponding transformation at inference.

*C*₈ Our approach involves the generation of image-based words for offline HWR to improve the training process of the offline HWR method with CMR techniques.

*C*₉ We evaluate DML techniques for DA between paper-based and tablet-based OnHW recognition systems.

*C*₁₀ We propose a dynamic negative sample selection technique for contrastive and triplet learning, which utilizes the Edit distance metric to measure the discrepancy between words. Our evaluations demonstrate improved classification accuracy for the proposed method in *C*₈ and *C*₉.

*C*₁₁ Our contribution is a benchmark for DA using AdaTime on 15 time-series datasets and 27 DML methods.

Regarding visual self-localization, our contributions can be summarized as follows:

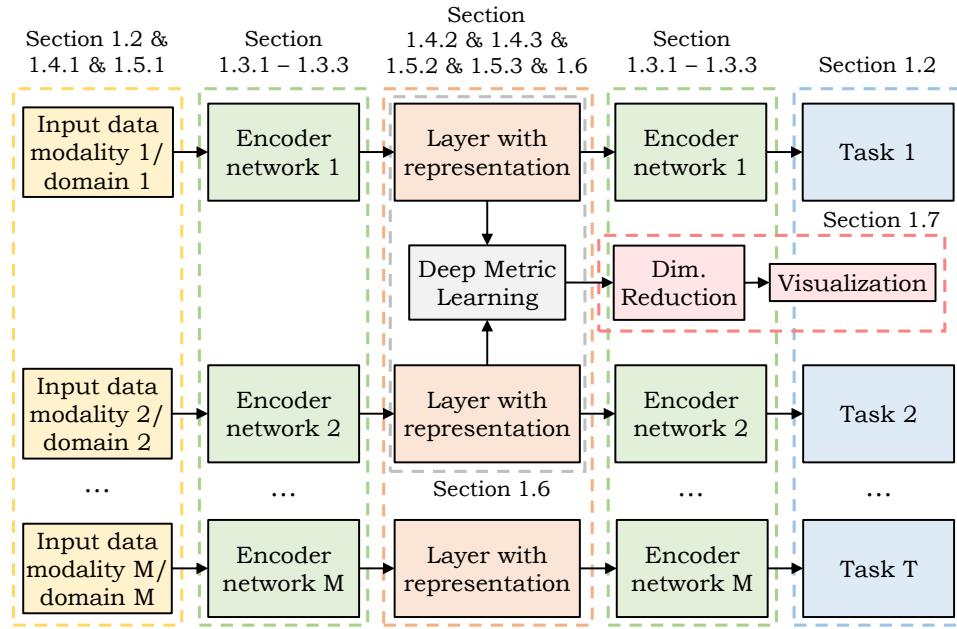


Figure 1.2: Structural overview of a cross-modal and multi-task neural network model, as detailed in the corresponding section. The model receives input data from M modalities/domains and solves T classification/regression tasks. For each modality, an encoder network extracts features. To align the representation between the networks, DML techniques are employed.

C₁₂ We collect and publish several datasets (the *Industry dataset*) at the Fraunhofer IIS L.I.N.K. test and application center. These datasets consist of visual and inertial sensor recordings along with corresponding ground truth poses. Specifically, scenario #2 was collected using a positioning system with wide-angle cameras, while scenario #3 was collected on a forklift truck that involved fast motion dynamics. Additionally, scenario #4 comprises visual and inertial data recordings from a hand-held setup and a small robotic system in a challenging large-scale indoor environment.

C₁₃ Our proposal involves a method for predicting absolute poses and relative poses from optical flow between corresponding images. We also introduce a fusion module that combines the predictions from absolute and relative pose estimation using recurrent units, resulting in improved performance on the absolute pose regression task. To enhance the generalizability, we conduct pre-training of both the absolute and relative models on simulated data.

C₁₄ Our study comprises a benchmark of the fusion between visual and inertial absolute and relative pose regression using CMR. We compare intermediate fusion, soft fusion, and late fusion through recurrent units. Additionally, we integrate auxiliary and Bayesian learning into the absolute pose regression task. We conduct experiments on both aerial vehicle and hand-held datasets, as well as on our Industry dataset.

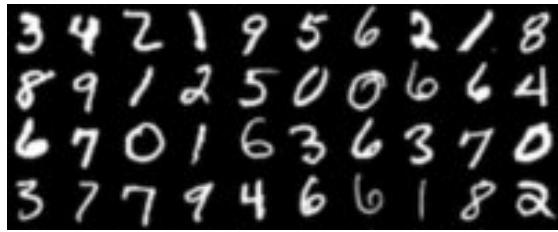
Table 1.1: Reference of the sections to the chapters of the corresponding, contributing paper.

Section	Contributing Paper	Contribution	Contributing Paper
1.2.1	2 – 9	C_1	3 & 6
1.2.2	10 & 11	C_2	3
1.3.1 – 1.3.3	2 – 11	C_3	6
1.3.4	2 & 4 & 7 – 11	C_4	4
1.4.1	4 & 8 & 10 & 11	C_5	6
1.4.2	8 & 10 & 11	C_6	5
1.4.3	8 & 9	C_7	7
1.5.1	5 & 7 & 9	$C_8 – C_{10}$	8 & 9
1.5.2 – 1.5.4	2 & 7	C_{11}	2
1.6	2 & 4 & 7 – 9	$C_{12} – C_{14}$	10 & 11 & 12
1.7	4 & 7 & 8 & 11		

In the following, we present an outlook of the thesis. Figure 1.2 illustrates a neural network with CMR and DML. The network is designed to predict $T \in \mathbb{N}$ classification or regression tasks ($T \geq 1$) by incorporating $M \in \mathbb{N}$ modalities/domains ($M \geq 2$), each with a domain shift between them. The features are extracted by M encoder networks, which are distinguished but not necessarily different. Each encoder’s layer output forms a representation, which is then utilized by another encoder network to predict the goal task by minimizing a loss function. The alignment of representations between different modalities is achieved using DML and contrastive or triplet learning. Dimensionality reduction is employed to visualize layer representations for evaluating the model’s performance. Our applications are limited to $M \leq 2$ modalities and $T \leq 2$ tasks. However, our methods can be extended to an arbitrary number of modalities and tasks. Table 1.1 provides an overview of the section and the contributions linked to the papers. Section 1.2 provides an overview of the applications OnHW recognition and visual self-localization. The corresponding modalities are defined in Section 1.4.1, and the domain shift is demonstrated in Section 1.5.1. In Section 1.2, for each application, we define the classification and regression tasks. We provide additional details regarding neural networks, specifically MTS-specific neural networks, in Sections 1.3.1 to 1.3.3. In Section 1.4, an overview of CMR and pairwise learning methods is presented based on the encoded feature representation, and Section 1.5 provides an overview of DA methods. Both CMR and DA rely on representation learning, which is mathematically defined in Section 1.6. Section 1.7 utilizes dimensionality reduction techniques to visualize feature embeddings. In Chapter 2, additional experiments are presented using the AdaTime toolbox. Chapters 3 to 12 comprises the contributing papers. Datasets and source code are publicly available for the OnHW¹ and self-localization² applications.

¹<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>
https://gitlab.cc-asp.fraunhofer.de/ottf/uq_time_series

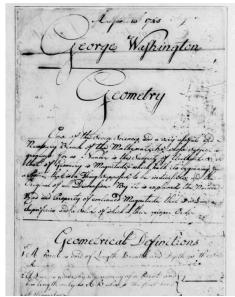
²https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets
<https://www.iis.fraunhofer.de/de/ff/lv/lok/opt1/warehouse.html>



(a) The MNIST dataset (LeCun et al., 1998).

In mid-april Anglesey
moved his family and
entourage from Rome to Naples,
there to await the arrival of

(b) The IAM-OffDB dataset (Marti & Bunke, 1999).



(c) The George Washington dataset (Fitzpatrick, 1931).



(d) The Devangari dataset (Acharya et al., 2015).

Figure 1.3: Exemplary datasets commonly used for offline HWR applications.

1.2 Applications

This section delves into the technical details and background of our OnHW recognition (in Section 1.2.1) and visual self-localization (in Section 1.2.2) applications. We introduce state-of-the-art and novel datasets, methods, loss functions, and the classification and regression tasks definitions. The background of our applications, along with the challenges and limitations highlighted, motivate the need for the methods proposed in the contributing papers.

1.2.1 Online Handwriting Recognition

First, we distinguish between offline HWR, which involves optical character recognition, and online trajectory-based HWR. Following this, we introduce OnHW recognition from sensor-enhanced pens used for writing on paper, and we demonstrate the shift in domains between writing on paper and touch screen surfaces, as well as between right-handed and left-handed writers. We then provide an outline of both contemporary and our cutting-edge datasets, as well as a definition of the classification tasks, loss functions, and trajectory regression tasks. Finally, we present an overview of HWR-based techniques.

Offline and Online HWR. The act of handwriting entails the creation of structured symbols that convey linguistic information, serving as a means of communication or docu-

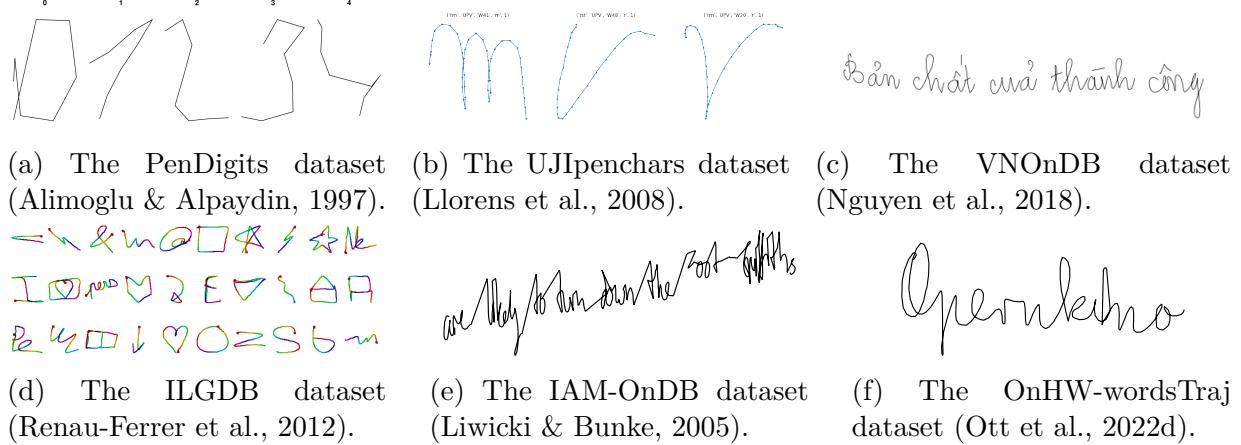


Figure 1.4: Exemplary datasets commonly used for online trajectory-based and gesture-based HWR applications. Note that data is given as 2D trajectories and not as images.

mentation. HWR refers to the process of digitizing handwritten text, which can be classified into offline and online HWR. While research on offline HWR has achieved a high level of sophistication and near-human performance, it is unsuitable for real-time recognition applications due to its unacceptable delay, as noted by Fahmy (2010). The recognition of handwritten text involves two consecutive processes, namely digitization and recognition. Figure 1.3 depicts some widely utilized data samples for offline HWR methods, although numerous other datasets are available. Optical character recognition exclusively focuses on the analysis of the visual representation of handwriting, and its efficacy is constrained since it fails to exploit temporal information such as writing direction, velocity, or pressure applied to the paper (Plamondon & Srihari, 2000). Online HWR, on the other hand, typically operates on various spatio-temporal signals such as the position of the pen tip (in 2D), its temporal context, or the movement on the writing surface. These signals can be partitioned into (indexed) strokes, for instance (Vinciarelli & Perrone, 2003). In contrast to offline HWR, online HWR (Ghosh et al., 2022) presents certain challenges, such as segmenting cursive written sequences into individual characters. Figure 1.4 illustrates some commonly recognized trajectory data used in online HWR methods. Numerous significant handwriting-related issues encountered in daily life necessitate an informative representation of the writing as well as effective classification algorithms (Hussain et al., 2015). Examples of such issues include signature verification, writer identification, and handwriting recognition. One well-established real-world application of online HWR involves the utilization of a stylus pen and a touch screen surface in various recording systems (Alimoglu & Alpaydin, 1997). These systems process the 2D position on the surface represented as a trajectory. Some exemplary setups include the iPad OS system and the reMarkable pad. There are two primary limits that afflict such systems. The first is that they necessitate a tablet with a touch screen surface and a stylus pen that has either integrated magnetometers or pressure sensitivity, and only function in conjunction with both. This results in costly applications that are insufficiently versatile. The second is that the techniques

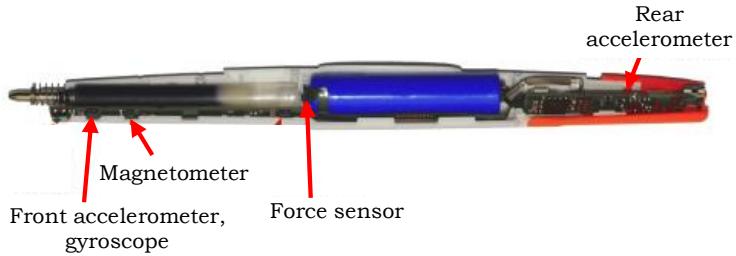


Figure 1.5: Sensor-enhanced pen with front and rear accelerometers, one gyroscope, one magnetometer, and one force sensor. Source: <https://stabilodigital.com/>.

employed in the iPad OS or the reMarkable necessitate writing on particular surfaces, which can influence the writing style. Additionally, some applications demand writing on regular paper, or the availability of a touch screen surface is not always assumed, such as when creating a short list but needing to digitize the notes later (Ott et al., 2023c). As a result, several advancements have been made in sensor-enhanced pens for writing on ordinary paper, which are detailed below.

Online HWR from Sensor-Enhanced Pens. Numerous prototype systems employing sensor-enhanced pens for OnHW recognition on normal paper have been developed (Wang et al., 2013; Toyozumi et al., 2016; Schrapel et al., 2018; Chen et al., 2021a; Bu et al., 2021; He et al., 2022a; Alemayoh et al., 2022; He et al., 2022b; Singh & Chaturvedi, 2023); however, they are not yet suitable for real-world applications. For instance, the GyroPen (Deselaers et al., 2015) simulates pen-like interaction using standard built-in sensors found in modern smartphones or the handwriting device by Kuramoto et al. (2020) uses a force-torque sensor. Drey et al. (2022) bypass a sensor-enhanced device by placing a paper on a tablet device. On the other hand, STABILO International GmbH offers a sensor pen that is a finished product and can be purchased. The pen comprises two accelerometers situated at the front and the back (each with 3 axes), a gyroscope (3 axes), a magnetometer (3 axes), and a force sensor (refer to Figure 1.5). All sensors capture data at a frequency of 100 Hz. The force sensor measures the pressure applied by the pen tip to the surface (i.e., paper). The data collection contains 14 measurements: four sensor data, each in x , y , and z direction (channels 1-12), the force sensor (channel 13), and the time step when the recording device receives the data from the pen. The recording device can be any Bluetooth-enabled device, such as smartphones, tablets, or computers. Dissimilar to OnHW recognition on touch screen surfaces, this sensor-enhanced pen permits an array of new and diverse applications:

1. *Graphomotor* skills are crucial for handwriting. The pen's visual feedback is advantageous for young pupils and children in language acquisition (Alonso, 2015; Wiley & Rapp, 2021; Drey et al., 2022). The device can assist in supporting school pupils' learning or self-direction learning from home without additional effort (Simonnet et al., 2018). Handwriting on paper is more beneficial for learning than typing on

a keyboard (Ihara et al., 2021) and has an impact on retention during note-taking (Wiechmann et al., 2022).

2. Sensor pens can aid in identifying Parkinson’s disease by detecting specific features from dynamic handwriting tests. From the raw signals, clinical diagnosis of the patient can be performed (Júnior et al., 2020; Kumar & Ghosh, 2023; Sarin et al., 2023).
3. Recognition of finger or hand air-writing in three-dimensional space is essential for virtual and augmented reality (Zhang et al., 2022). Smart devices can substitute previously used optical sensors. Optical sensor-based methods are restricted by lighting conditions that significantly affect the recognition result.
4. Air-writing applications are related to controlling smart devices such as television sets, roller blinds, etc., while simultaneously having a pen quickly jotting down notes.
5. Writer identification finds its application in forensics or security (Christlein et al., 2015).
6. An examination of gender differences in kinematic, dynamic, and temporal features was conducted by Faundez-Zanuy & Mekyska (2023). The study discovered that OnHW recognition studies need to consider gender as a significant confounding factor. A gender disparity is apparent in the accelerometer data of text written in capital letters, on-surface and in-air time of genuine and forged signatures, and the writing time in cursive characters.

Datasets. The primary focus of the contributing papers was to advance research in handwriting utilizing sensor-enhanced pens, specifically by recording a range of single character and sequence-to-sequence datasets using the pen, resulting in the creation of the *online handwriting (OnHW)* database (Ott et al., 2020b, 2022d). Table 1.2 provides a summary of all the OnHW datasets and compares them to the typical trajectory-based datasets for writer-dependent (WD) and writer-independent (WI) classification tasks. The OnHW-chars dataset consists of 31,275 samples of single lowercase (‘a’, ‘b’, …, ‘z’) and uppercase (‘A’, ‘B’, …, ‘Z’) characters from 52 classes, collected from 119 writers. The OnHW-symbols dataset includes 2,327 samples of 10 number classes (‘0’, ‘1’, …, ‘9’) and 5 operator classes (‘+’, ‘-’, ‘.’, ‘:’, ‘=’) from 27 writers. Left-handed writers are also included in both single character datasets, which are denoted as OnHW-chars-L and OnHW-symbols-L. The OnHW-equations dataset is a sequence-based equations dataset consisting of 10 numbers and 5 operators written by 55 writers, which was included in the *UbiComp 2021 challenge*³. The OnHW-words500 and OnHW-wordsRandom datasets contain more word samples (25,218 samples and 14,641 samples, respectively). The OnHW-words500 dataset contains the same 500 words for each writer, while the OnHW-wordsRandom dataset contains randomly chosen samples from a large word list in both

³UbiComp 2021 challenge: <https://ubicomp.org/ubicomp2021/cfp/student-challenges-2/>

Table 1.2: Overview of the OnHW datasets (right-handed and left-handed) and state-of-the-art online handwriting datasets for writer-dependent (WD) and writer-independent (WI) tasks. Source: Ott et al. (2022d).

Dataset	Number Writers	Number Classes	Maximal Length	Total	Number Samples		Total Chars.
					WD	WI	
OnHW-chars	119	52	single	31,275	23,059	8,216	31,275
OnHW-chars-L	9	52	single	2,270	1,816	454	-
OnHW-symbols	27	15	single	2,326	1,853	473	2,326
OnHW-symbols-L	4	15	single	361	289	72	361
OnHW-equations	55	15	15	10,713	8,595	2,118	106,968
OnHW-equations-L	4	15	15	843	677	166	8,438
OnHW-words500(R)	53	59	19	25,218	20,176	5,042	137,219
OnHW-words500-L	2	59	19	1,000	800	200	5,438
OnHW-wordsRandom	54	59	27	14,641	11,744	2,897	146,350
OnHW-wordsRandom-L	2	59	26	996	798	198	10,029
OnHW-wordsTraj	2	59	10	16,752	13,250	3,502	146,512
ICROW (Schomaker, 2003)	67	53	15	13,119	10,500	2,619	90,138
IAM-OnDB (Liwicki & Bunke, 2005)	197	81	64	10,773	8,702	2,071	265,477
VNOnDB-words (Nguyen et al., 2018)	201	147	11	110,746	88,677	22,069	368,455

German and English. Although the OnHW-equations dataset contains a smaller number of samples (10,713), it has a larger average sequence length compared to the other datasets. The OnHW-wordsTraj dataset is obtained by utilizing a Wacom EMR module to record trajectories on a Samsung Galaxy Tab S4, in place of the ink refill. The dataset is comprised of four distinct data sources: (1) The actual trajectory of the pen tip on the tablet, which serves as the ground truth and is recorded at a frequency of 30 Hz, (2) the sensor data collected by the sensor-enhanced pen, (3) and four cameras are set up to capture the movement of the pen tip at 60 Hz. We manually label the pixels of 100 random images to predict the pen tip pixels and obtain the pen tip trajectory in camera coordinates. (4) The corresponding word labels are also provided. The ground truth trajectory data in this dataset is comparable to that of the ICROW (Schomaker, 2003), IAM-OnDB (Liwicki & Bunke, 2005), and VNOnDB-words (Nguyen et al., 2018) datasets. Please compare Figure 1.4c, Figure 1.4e, and Figure 1.4f. The IAM-OnDB (line level) and VNOnDB-words datasets contain a greater number of samples, but also require a larger training dataset due to the higher number of character classes (81 and 147, respectively). The challenges of the OnHW dataset are presented in Figure 1.6 where sensor data for different channels from four different datasets are displayed. Figure 1.7 shows all sensor channels for several samples, as well as the mean and standard deviation of the OnHW-words500 and OnHW-equations datasets, to highlight the variance.

Challenges. Our datasets and evaluations constitute a benchmark foundation for the development of novel methodologies and present important challenges for future research, which we will outline in the following:

1. To cover the large variety of possible holding options of the pen tip and the different rotation and movement patterns that can occur due to the use of a gyroscope and

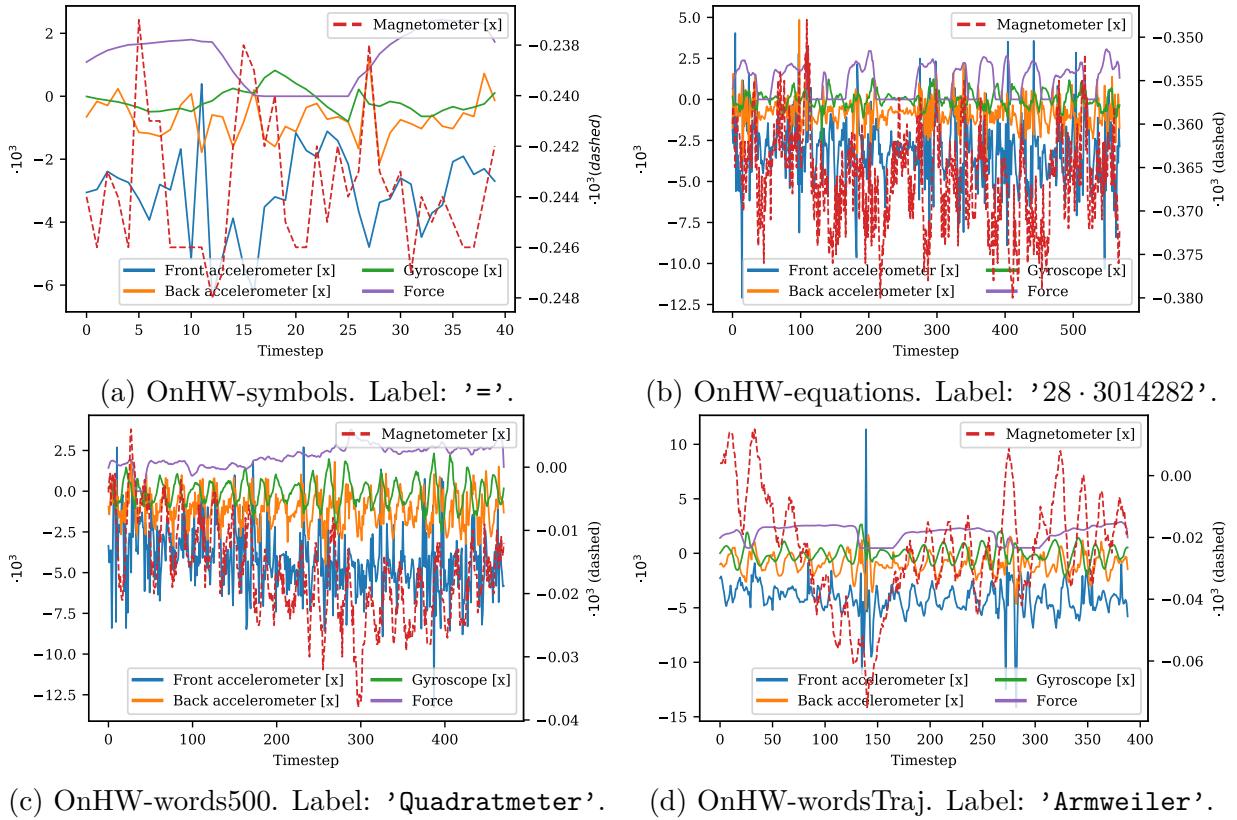


Figure 1.6: Exemplary sensor data of handwritten symbols, words, and equations extracted from the OnHW datasets are presented. Note that the force signal drops to 0 for symbols and equations. As a result, numbers and symbols can be segmented into single strokes, and equations can be segmented into individual numbers, symbols, or strokes.

accelerometer, a large number of samples has to be recorded to account for differences between beginner and advanced writers.

2. The variability of magnetometer data is affected by the writing surface and the proximity of a magnetic device to the pen. This can be observed in the difference in spread of magnetometer data in the x -axis between words (a) and equations (b) in Figure 1.7, where words have a higher variance.
3. The variance in writing style, such as cursive or printed writing, differs among individuals. This is demonstrated in Figure 1.7 where the gyroscope samples in the y -direction show a higher variance for words compared to equations, as equations are usually written in a printed style.
4. A character is composed of multiple strokes that can be written in various directions and ordered in a non-uniform manner.
5. Sequences of characters can be written separately, then the characters and strokes

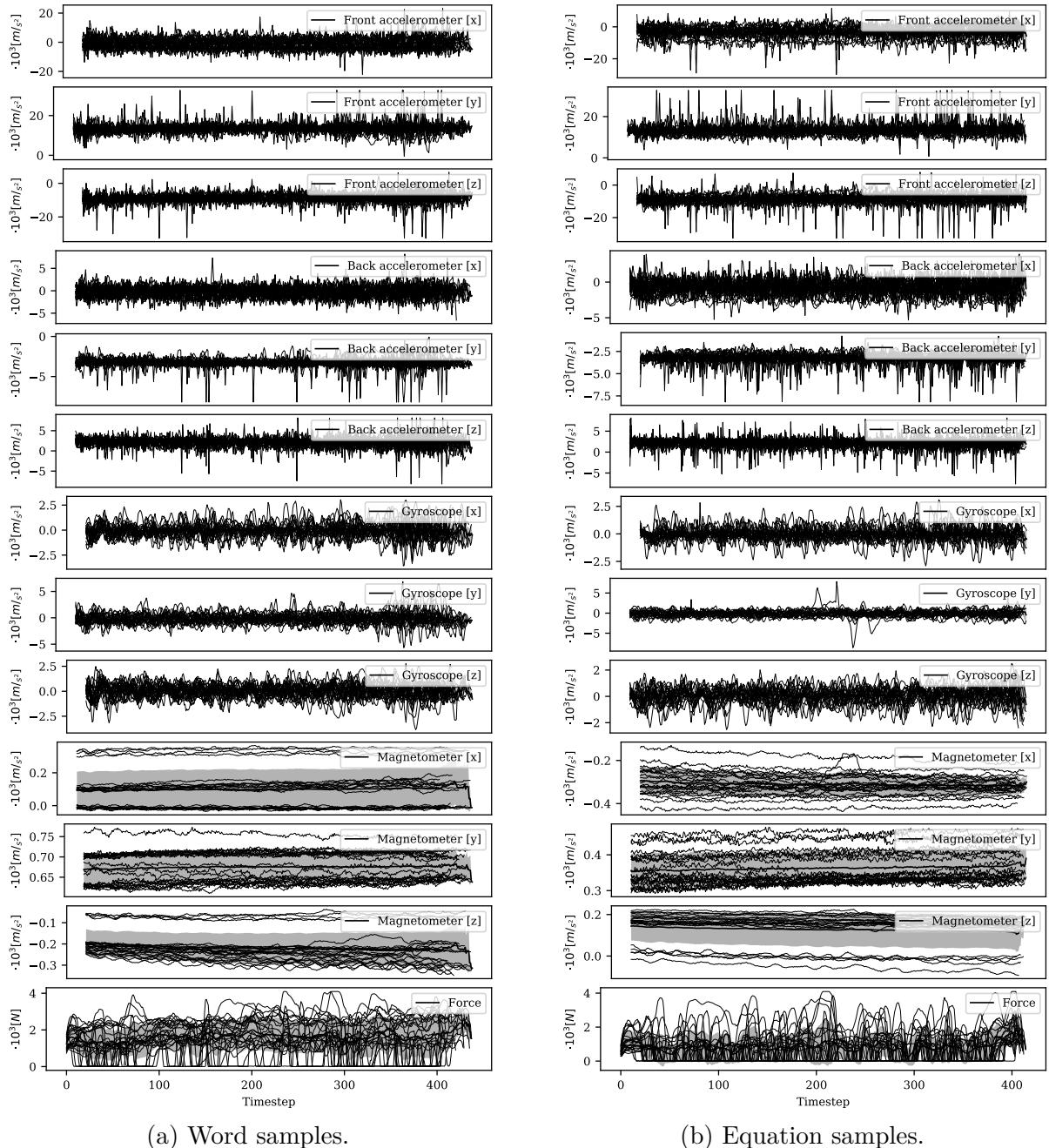


Figure 1.7: Exemplary sensor signals (mean and standard deviation) for written words and equations for the front and rear accelerometer, gyroscope, magnetometer (each of 3 axis), and the force sensor integrated in the sensor-enhanced pen.

can be segmented and classified in a straightforward way, or they can be written in a cursive style, making it necessary to classify sequences using a sequence-based loss. This can be seen by comparing the equation in Figure 1.6b and the word in

Figure 1.6c. The cursive writing style also results in a higher average force, as shown in the last row of Figure 1.7.

6. The sensor data is affected by the microscopic elevations of the paper, which introduce noise and reduce model performances, as can be seen by comparing Figure 1.6c and Figure 1.6d).
7. Another challenge related to the large variety of possible holding options (point 1) of the pen tip is the domain shift between sensor data (i.e., gyroscope) from right-handed and left-handed writers. As left-handed writers are an under-represented group in the real-world (constituting only 10.6%), ML models tend to overfit on right-handed writers, leading to decreased performance on left-handed writers.
8. The noise level in the sensor data for writing on paper and tablet is significantly different, resulting in a domain shift between the two, which is related to point 6. Combining samples from both domains cannot be done straightforward, as it can lead to a decrease in model performance. Figure 1.8 illustrates this issue.

Domain Shift for Writing on Paper & Tablet. In light of the existence of numerous setups that utilize stylus pens in conjunction with tablets, as well as a variety of sensor-enhanced pens designed for use with standard paper, the following inquiry arises: *Is it feasible to integrate these two approaches, specifically by utilizing a sensor-enhanced pen with an interchangeable pen tip for writing on both paper and tablet?* Such an integration could potentially combine a multitude of applications in a versatile manner. Therefore, we conduct a detailed analysis of the domain shift of sensor data between writing on paper and tablet (see Figure 1.8). Upon comparing the time-series of the front accelerometer in the x -, y -, and z -directions (rows 1 to 3), it is evident that the sensor data for writing on paper exhibits substantially more noise than that for writing on a tablet. Although the noise is reduced for the rear accelerometer (rows 4 to 6), the difference remains discernible. For the gyroscope data, there is no disparity in noise; however, the writing frequency is higher for writing on a tablet as the smooth surface allows for an increase in writing velocity. In addition, it should be noted that the magnetometer data corresponding to rows 10 to 12 exhibit varying levels of dispersion across the scaling range. Specifically, there is a discernible positive trend in the y -axis of the magnetometer data, which may be attributed to the left-to-right writing direction on the tablet, and the resultant fluctuations in the magnetic field. It is worth mentioning, however, that the force data remains unchanged. Readers interested in mitigating domain shift via DA and representation learning methods are directed to Chapter 9.

Domain Shift for Right-Handed & Left-Handed Writers. Detecting a domain shift through data analysis can be a sophisticated task as it may not capture the domain shift present in the model. However, uncertainty quantification techniques can be employed to estimate both data and model uncertainty, thereby providing an indicator of domain

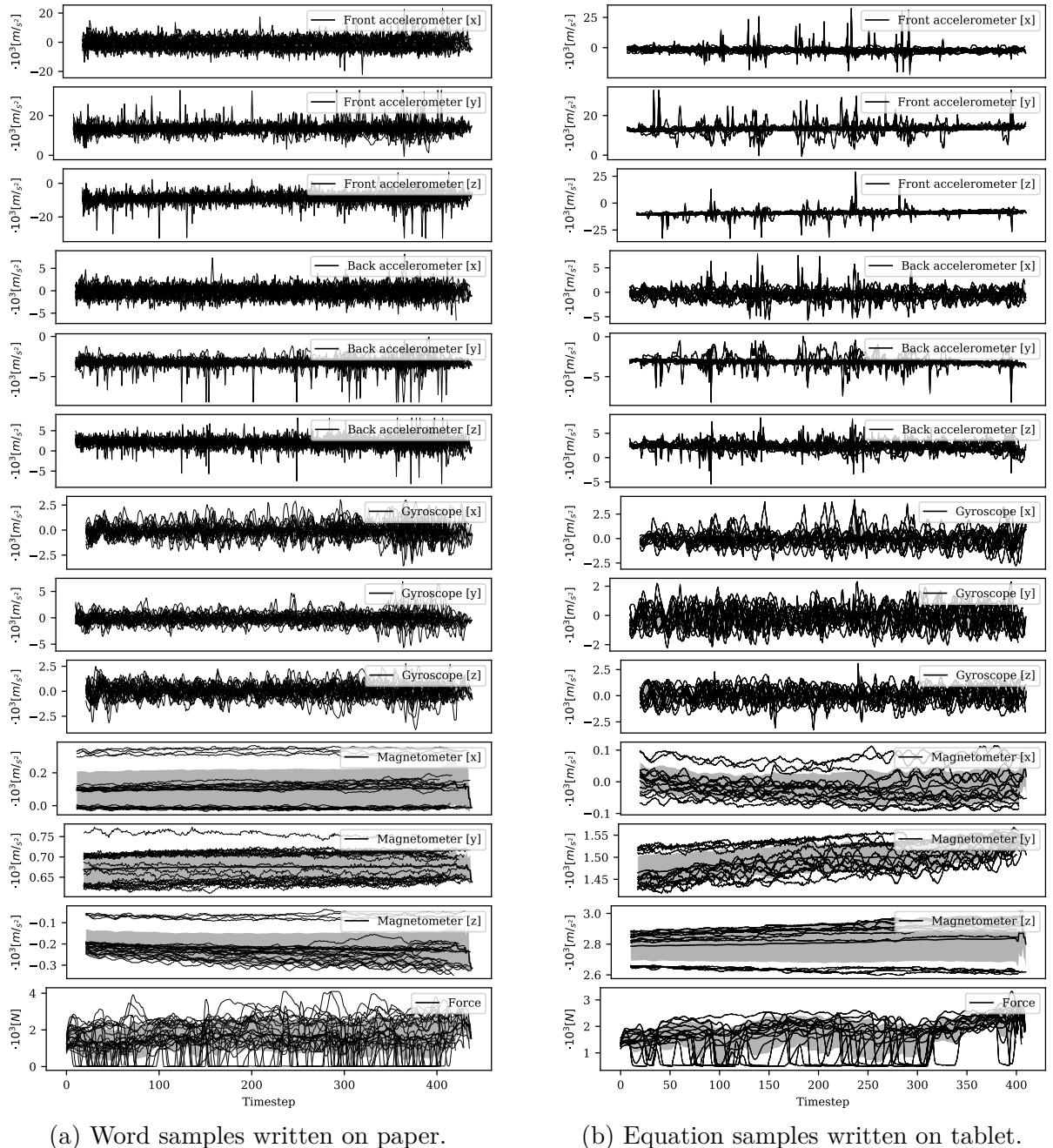


Figure 1.8: Exemplary sensor signals (mean and standard deviation) for words written on paper and tablet for the front and rear accelerometer, gyroscope, magnetometer (each of 3 axis), and force sensor integrated in the sensor-enhanced pen. Note that we plot 24 samples for writing on paper and 53 samples for writing on tablet.

shift. While there exist advanced uncertainty quantification methods for computer vision applications, such methods are seldom employed for spatio-temporal data, such as OnHW

data. According to Jospin et al. (2022), the total uncertainty can be decomposed into two distinct components: the *aleatoric* (data) and *epistemic* (model) uncertainty. The aleatoric uncertainty refers to the inherent stochastic uncertainty due to noise and errors, which cannot be eliminated. When it comes to OnHW recognition, the source of such uncertainty include shaky hands of the writers, noisy sensors, unsteady writing behavior, and similar-looking characters. On the other hand, the epistemic uncertainty stems from systemic uncertainty and could potentially be reduced with improved models or more data. In OnHW recognition, potential sources of uncertainty include new writing styles during testing that were not present in the training dataset or a lack of similar pen-holding styles in the training set. To assess the uncertainty in OnHW data, see Klaß et al. (2022), we conducted a comprehensive evaluation using two widely-used Bayesian inference techniques: The first technique is Stochastic Weight Averaging-Gaussian (SWAG) proposed by Maddox et al. (2019), which involves computing the average of stochastic gradient descent iterates to gain insight into the geometry of the posterior distribution $p(\theta, D)$ over model parameters θ , given the training dataset D . This posterior is approximated using a Gaussian distribution. The second technique is Deep Ensembles introduced by Lakshminarayanan et al. (2017) that refers to a committee of neural networks, each initialized with a different seed to introduce stochasticity. For an in-depth analysis of the model uncertainty of both right-handed and left-handed writers, please refer to Chapter 5. One potential solution to the uncertainty arising from new writers is to employ domain adaptation, which is also applicable to offline HWR, as demonstrated by Kohút & Hradiš (2023) who found that fine-tuning with data augmentation can effectively address the challenge of DA in HWR.

Definition of the Classification Tasks. In the following, we present the notation used for the OnHW recognition task, which is treated as a classification problem and remains consistent across the relevant papers (Ott et al., 2020b, 2022a,b,c,d; Klaß et al., 2022; Ott et al., 2023c). One single character, symbol or number, or a sequence of characters (i.e., words or equations) is represented as an MTS of the 13 sensor channels provided by the pen. An MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $l \in \mathbb{N}$ (here, $l = 13$) streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l}), i \in \{1, \dots, m\}$, where $m \in \mathbb{N}$ is the length of the time-series. For the OnHW recordings, the lengths of the time-series are variable. For the single character classification task, we interpolate the time-series to $m = 64$ for the OnHW-chars dataset and to $m = 79$ for the split OnHW-equations and OnHW-symbols dataset. For a study on time-series length of sequences, see Ott et al. (2022d). The MTS training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\} \in \mathbb{R}^{n_U \times m \times l}$, where n_U is the number of time-series. Each MTS is associated with v for single class prediction and \mathbf{v} for sequence-based prediction, where \mathbf{v} is a sequence of L class labels from a pre-defined label set Ω with K classes ($L = 1$ for single class prediction). The aim of MTS classification is to predict an unknown class label $v \in \Omega$ for a given MTS for the single character classification, and to predict a sequence of unknown class labels $\mathbf{v} \in \Omega^L$. The labels $\mathcal{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_{n_U}\} \in \Omega^{n_U \times L}$ are the corresponding training labels to the training MTS set \mathcal{U} .

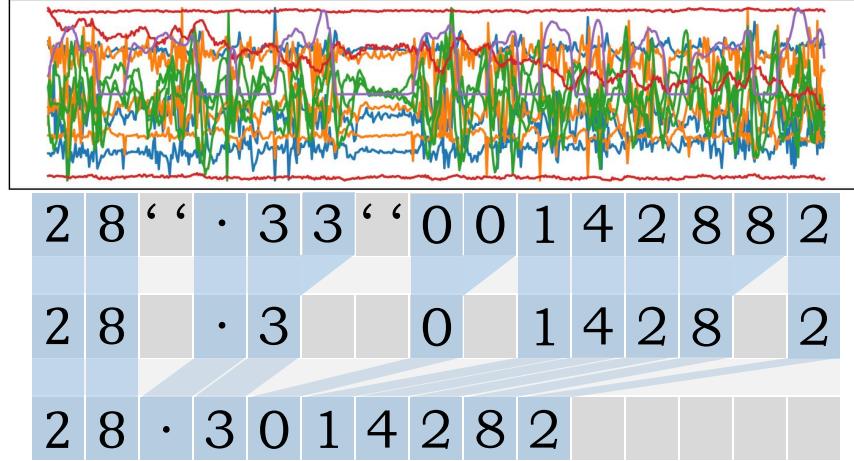


Figure 1.9: Visualization of the CTC loss function applied to the prediction of a sequence of numbers and symbols from time-series data of a single sample belonging to the OnHW-equations dataset. Label: '28 · 3014282', blank label: {' '}'.

Loss Functions. For the single classification task, we use the categorical cross-entropy (CE) loss defined by

$$\mathcal{L}_{\text{CE}}(\mathcal{U}, \mathcal{V}) = -\frac{1}{K} \sum_{i=1}^K q(i|\mathbf{u}) \log p(i|\mathbf{u}), \quad (1.1)$$

where $p(i|\mathbf{u})$ is the predicted probability for the i^{th} class and $q(i|\mathbf{u})$ is the true class distribution. K are the number of classes. For the single character classification task (lowercase and uppercase combined), it holds $K = 52$. For the classification of symbols and numbers combined, it holds $K = 15$. In Ott et al. (2022d), we use variants of the CE loss to reduce overfitting to noisy or unbalanced labels, such as the focal loss (Lin et al., 2017), label smoothing (Pereyra et al., 2017), soft and hard bootstrapping loss functions (Reed et al., 2015), generalized CE (Zhang & Sabuncu, 2018), symmetric and reverse CE (Wang et al., 2019), and joint optimization (Tanaka et al., 2018). For all sequence-based classification tasks, we use the connectionist temporal classification (CTC) loss function $\mathcal{L}_{\text{CTC}}(\mathcal{U}, \mathcal{V})$. This practice is consistent across all contributing papers (Ott et al., 2022b,d, 2023c). For completeness, we mathematically define the CTC loss function, introduced in detail by Graves et al. (2006), in the following. An exemplary visualization of the CTC loss on a sample equation is presented in Figure 1.9. The output layer of our architecture consists of a softmax function with $|L|$ units with one additional unit, where $L = \Omega$, and where Ω represents the set of class labels. It holds $L' = L \cup \{' '\}$. The activation value of the additional unit in the softmax output layer corresponds to the probability of observing no label {' '}. Hence, for an input sequence \mathbf{x} of length T , a continuous map $\mathcal{N}_{\mathbf{w}}$ (with weight vector \mathbf{w}) from the input $(\mathbb{R}^m)^T$ to the output $(\mathbb{R}^n)^T$, and the sequence of network outputs $\mathbf{v} = \mathcal{N}_{\mathbf{w}}(\mathbf{x})$, the activation of output unit k at time t is v_k^t and it holds the probability $p(\pi|\mathbf{x}) = \prod_{t=1}^T v_{\pi_t}^t$, $\forall \pi \in L'^T$, which defines a distribution over the set L'^T of length T sequences. We define the conditional probability of a given labelling $\mathbf{t} \in \Omega^{\leq T}$

as $p(\mathbf{t}|\mathbf{x}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{t})} p(\pi|\mathbf{x})$ with the many-to-one map \mathcal{B} and set of possible labels $\Omega^{\leq T}$. Then, the objective of the classifier is to predict the most probable labelling for the input sequence that is defined by $h(\mathbf{x}) = \arg \max_{\mathbf{t} \in \Omega^{\leq T}} p(\mathbf{t}|\mathbf{x})$ that corresponds to the most probable path $h(\mathbf{x}) \approx \mathcal{B}(\pi^*)$ where the optimal path is $\pi^* = \arg \max_{\pi \in \mathcal{N}^T} p(\pi|\mathbf{x})$. For more information, refer to Graves et al. (2006).

Evaluation Metrics. In the following, we introduce a set of task-specific metrics for sequence-to-sequence and single character-based evaluations, which are frequently employed and used throughout in our papers. To ensure consistency across our evaluations, we report the evaluation metrics and the Edit distance (ED). To evaluate single character recognition, we use the character recognition rate (CRR), defined as the ratio of correctly classified characters to the total number of character in the test set, refer to Ott et al. (2020b, 2022a,c,d); Klaß et al. (2022). For sequence-to-sequence evaluation, we use the character error rate (CER) and word error rate (WER) as metrics, both of which are based on the ED. The ED is the minimum number of substitutions S , insertions I , and deletions D required to transform sequence $\mathbf{f} = (f_1, \dots, f_r)$ of length r into sequence $\mathbf{g} = (g_1, \dots, g_n)$ of length n . The ED is defined by

$$ED_{i,j} = \begin{cases} ED_{i-1,j-1} & \text{for } f_i = g_j \\ \min \begin{cases} ED_{i-1,j} + D(f_i) \\ ED_{i,j-1} + I(g_j) \\ ED_{i-1,j-1} + S(f_i, g_j) \end{cases} & \text{for } f_i \neq g_j \end{cases} \quad (1.2)$$

for $1 \leq i \leq r, 1 \leq j \leq n$, $ED_{i,0} = \sum_{k=1}^i D(f_k)$ for $1 \leq i \leq r$, and $ED_{0,j} = \sum_{k=1}^j I(g_k)$ for $1 \leq j \leq n$ (Damerau, 1964; Ott et al., 2022d). We define the CER = $\frac{S_c + I_c + D_c}{N_c}$ as the ED, the sum of character substitutions S_c , insertions I_c , and deletions D_c , divided by the total number of characters in the set N_c . Similarly, the WER = $\frac{S_w + I_w + D_w}{N_w}$ is computed with word operations S_w , I_w , and D_w , and number of words in the set N_w (Kang et al., 2022), refer to Ott et al. (2022b,d, 2023c).

Definition of the Trajectory Regression Task. In some applications, it is necessary to capture the trajectory of the pen tip, in addition to/or instead of performing classification tasks. This is often required for tasks such as handwriting analysis or signature verification. However, accurately capturing the pen trajectory using inertial sensors is challenging due to the presence of both rotational and translational motion, which can degrade the accuracy of the captured data. Additionally, the pen tip motion between adjacent strokes in the air can result in off-plane traces, which must be removed from the final trajectory (Bu et al., 2021). To formally define the trajectory regression task, we consider a (discrete) MTS of varying length that takes values in $\Psi \in \mathbb{R}$ of size $n \times d$, where n is the number of time steps and d is the dimension of the time-series, typically two-dimensional. The objective is to predict a time-series $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d}$ that closely approximates the ground truth time-series $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_m\} \in \mathbb{R}^{m \times d}$. In the OnHW trajectory regression task, the ground truth

trajectory is obtained using a Samsung Galaxy Tab S4 with dimensions of $(100, 2)$. The objective is to minimize the distance $\mathbf{d}_i = \mathbf{y}_i - \mathbf{x}_i$ between the predicted trajectory \mathcal{X} and the ground truth trajectory \mathcal{Y} by minimizing the (differentiable) loss function $\mathcal{L} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$. In the contributing paper (Ott et al., 2022c) (refer to Chapter 4), we utilize distance-based, spatio-temporal, and distribution-based loss functions (refer to Section 1.6).

Methods & Architectures. Numerous techniques are available for offline HWR, as outlined in Chapter 8. In our paper contribution (Ott et al., 2023c), we opted gated text recognizer (GTR), as presented by Yousef et al. (2020). Several recent methods have also been proposed, including those by Diaz et al. (2021); de Sousa Neto et al. (2022); Bautista & Atienza (2022); Lyu et al. (2022); Chaudhary & Bali (2022); Jangpangi et al. (2022); Li et al. (2023). In general, any time-series classification method can be utilized for online HWR, provided it can be extended with the CTC loss for sequence prediction. For a comprehensive overview, we direct the reader to Chapter 6. We conducted evaluations on recurrent units (Chung et al., 2014), BiLSTMs, TCN (Bai et al., 2018), FCN (Wang et al., 2016), a combination of recurrent units with FCN (Karim et al., 2017, 2019; Elsayed et al., 2019), ResCNN (Zou et al., 2019), ResNet (Wang et al., 2016), XResNet (He et al., 2019), InceptionTime (Fawaz et al., 2020), XceptionTime (Rahimian et al., 2019), Transformer models (Zerveas et al., 2021), TapNet (Zhang et al., 2020b), and XEM (Fauvel et al., 2022) for the single character-based and sequence-to-sequence classification tasks. Further details can be found in Section 1.3.3. For OnHW recognition in DA and CMR applications, we consistently employed a combination of CNN and BiLSTM networks.

1.2.2 Visual Self-Localization

We first provide a rationale for employing visual self-localization and provide a definition of the pose regression tasks. Subsequently, we demarcate our proposed datasets from the current leading datasets and present a summary of the pose regression methods. Finally, we succinctly outline our methodological contributions.

Motivation & Challenges. The objective of self-localization is to determine the pose of an object in an arbitrary environment, which comprises its six degree-of-freedom position and orientation. This is essential in various applications, including:

1. In robotics, self-localization is important to navigate robots in their environment.
2. The pose of an autonomous vehicle is critical to accurately navigate in the environment and avoid collisions.
3. For augmented reality, self-localization is necessary to track the location of objects and overlay information onto them.
4. For industrial automation, utilizing self-localization technology can enable the identification of the precise pose of components on an assembly line, thus enhancing efficiency and productivity.

Depending on the applications, employing different sensor infrastructures such as LiDAR, radio, or radar-based systems can be advantageous (Cabrera-Ponce et al., 2021; Yu et al., 2022). Visual self-localization, on the other hand, leverages monocular or stereographic images to predict the pose (Kendall et al., 2015; Radwan et al., 2018). Inertial self-localization has the benefit of utilizing low-cost inertial measurement unit (IMU) sensors (do Monte Lima et al., 2019). However, for many localization tasks, achieving highly accurate pose estimation is a requirement, which is influenced by the chosen infrastructure or sensor setup. Novel techniques utilize two or more sensors to exploit the advantages of each sensor and attain improved pose prediction, resulting in lower pose error. Combining multiple sensors, i.e., different modalities, necessitates (1) addressing the challenges of each sensor with respect to the environment, and (2) developing a method that optimally fuses information from all modalities. In the contributing papers (Ott et al., 2020a, 2023a,b), we focus on self-localization utilizing visual and inertial sensors and tackle the challenges in the following:

1. Stereographic cameras are relatively expensive compared to monocular cameras, which are a low-cost alternative.
2. Visual self-localization is susceptible to the presence of repetitive or texture-less patterns in the environment, which can lead to performance degradation.
3. The prediction of pose in visual self-localization can be adversely affected by motion blur caused by fast-moving object in the scene.
4. The accuracy of pose prediction in self-localization can be significantly impacted by the scale (small-scale versus large-scale) of the environment, i.e., the distance of features from the object being localized.
5. Methods that rely on predicting the relative pose between two consecutive images or from inertial data can be susceptible to long-term drift, which may be induced by sensor noise.

Our primary objective is to improve the pose prediction accuracy using monocular images by leveraging additional sensor, modality information, or auxiliary tasks. In the following, we outline the different pose regression tasks that serve as baseline models for our cross-modal techniques.

Pose Regression Tasks & Loss Functions. Pose regression techniques have developed into efficient and high-performing methods for self-localization, which rely on convolutional (as discussed in Section 1.3.1) or recurrent (as discussed in Section 1.3.2) networks. Absolute pose regression (APR) techniques can regress the absolute pose (comprising of position and orientation) directly from image inputs. To refer to the abbreviated term for APR from images, we use APR_V. Typically, APR_V methods minimize the loss function

$$\mathcal{L}_{\text{APR}} = \|\hat{\mathbf{p}} - \mathbf{p}\|_2 + \alpha \left\| \hat{\mathbf{q}} - \frac{\mathbf{q}}{\|\mathbf{q}\|} \right\|_2, \quad (1.3)$$

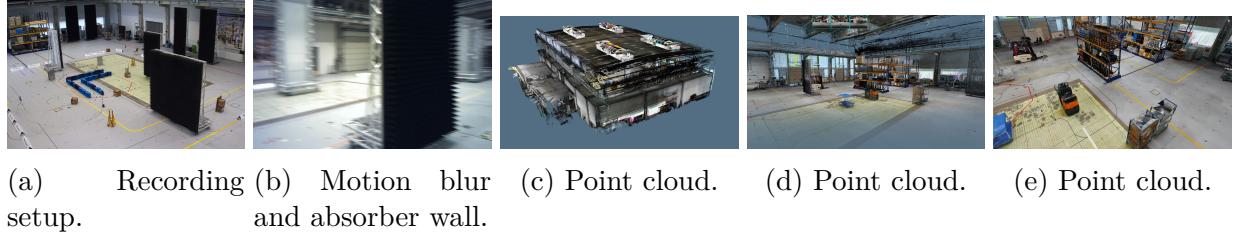


Figure 1.10: Visualization of the recording setup, challenges, and the point cloud in the Fraunhofer IIS L.I.N.K. test and application center.

where $\mathbf{p} \in \mathbb{R}^3$ is the predicted position, $\hat{\mathbf{p}}$ is the ground truth position, $\mathbf{q} \in \mathbb{R}^4$ is the predicted orientation represented as a quaternion, and $\hat{\mathbf{q}}$ is the ground truth orientation (Kendall et al., 2015). $\|\cdot\|_2$ is the Euclidean distance (L_2 norm). The hyperparameter α is responsible for the weighting of the loss terms and is contingent on the environment’s scaling. The absolute pose is denoted as $\mathbf{x} = [\mathbf{p}, \mathbf{q}]$. While odometry techniques rely on classical approaches for predicting the relative pose, contemporary relative pose regression (RPR) methods use CNNs or RNNs to regress the relative pose. We use the abbreviation RPR_V to denote RPR techniques based on image inputs, and RPR_I to denote those based on IMU inputs. Analogous to the loss function presented in Equation 1.3 for APR, we perform regression on the relative pose using:

$$\mathcal{L}_{\text{RPR}} = \|\Delta\hat{\mathbf{p}} - \Delta\mathbf{p}\|_2 + \beta \left\| \Delta\hat{\mathbf{q}} - \frac{\Delta\mathbf{q}}{\|\Delta\mathbf{q}\|} \right\|_2, \quad (1.4)$$

where $\Delta\mathbf{p}$ is the relative position (i.e., translation) and $\Delta\mathbf{q}$ is the relative orientation (i.e., rotation). $\Delta\hat{\mathbf{p}}$ and $\Delta\hat{\mathbf{q}}$ are the relative ground truth position and orientation, respectively. The hyperparameter β is used to weight the loss. We discuss the tasks APR_V, RPR_V, and RPR_I in the respective contributing papers (Ott et al., 2020a, 2023a,b). In the following, we present an overview of the datasets for evaluating our proposed techniques.

Contributing Datasets. Our objective is to achieve self-localization of an object, specifically an autonomous robot, forklift truck, or human, within large-scale industrial environments. Although there are publicly accessible datasets like Microsoft 7-Scenes (Shotton et al., 2013), Cambridge Landmarks (Kendall et al., 2015), DeepLoc (Radwan et al., 2018), and KITTI (Geiger et al., 2013), they fail to satisfy our requirements of (1) utilizing only monocular and IMU data sources, (2) providing highly accurate ground truth poses of less than 1 cm, and (3) encompassing large-scale industrial environments. As a result, we record the Industry dataset captured in the Fraunhofer IIS L.I.N.K. test and application center, which features four different scenarios with diverse recording setups (refer to Figure 1.10a for an exemplary setup). The visual-only Industry scenario #1 dataset was recorded by Löffler et al. (2018) using a positioning system. The dataset comprises eight testing datasets, which can be employed to assess volatility, scale transition, and generalizability. We tested the approach presented in Ott et al. (2020a) on the Industry scenario

#1, scenario #2, and scenario #3 datasets. While we recorded the Industry scenario #2 using a positioning system, we used wide-angle cameras instead of narrow-angle images from scenario #1. For scenario #3, we employed a wide-angle camera setup on a forklift truck to assess the challenging motion blur (Alam et al., 2023) caused by fast-moving dynamics (see Figure 1.10b). The visual-inertial Industry scenario #4 dataset is the most extensive scenario in size. It was captured using a small robotic system that integrates one Orbbec3D camera (23 Hz) and an IMU (140 Hz) and comprises of eight training trajectories and eleven testing trajectories. Environmental setups were altered between each recording by adding none to five large absorber walls, a labyrinth, or random objects in the environments. This variation enables the evaluation of the models’ generalizability on various environments. In two of the training and testing trajectories, we allowed two individuals to walk in the L.I.N.K. test and application center with the same recording setup, and we evaluated this recording in the contributing paper (Ott et al., 2023b). Furthermore, we conduct an evaluation on all datasets of Industry scenario #4 in the contributing paper (Ott et al., 2023a). Furthermore, we recorded a point cloud (see Figure 1.10c) with a NavVis M4 system to load this point cloud into a simulation framework (see Figure 1.10d and 1.10e) to generate an arbitrary number of synthetic images.

Contributing Methods. In the following, we provide a summary of our principal contributions on visual-inertial self-localization. Our primary model, denoted as APR_V, estimates the absolute (global) poses using image inputs. As a baseline model, we employ a time-distributed PoseNet (Kendall et al., 2015) trained with the loss function in Equation 1.3. However, we observe that the accuracy of pose predictions may be considerably affected by certain image properties, such as blur or noise caused by fast movements or environmental challenges such as repetitive or texture-less patterns. In such scenarios, the objective is to augment the absolute pose prediction and thereby reduce the error of the absolute pose by incorporating auxiliary information. In our prior work (Ott et al., 2020a, 2023a), we employed relative (local) poses from visual inputs, whereas in our subsequent work (Ott et al., 2023b), we used relative poses from inertial inputs. The additional knowledge of the relative pose between two successive global poses helps in reducing the absolute pose outliers and consequently, smooths the trajectory estimations. However, combining absolute and relative poses is a challenging task as the relative model only contains information about the change in pose relative to the preceding pose. Consequently, learning the relationship between the relative pose and absolute pose becomes challenging. To tackle this challenge, we explore various fusion techniques (refer to Table 1.3). In our previous work (Ott et al., 2020a), we estimated the absolute pose by employing a time-distributed PoseNet (Kendall et al., 2015) and computing the optical flow between two consecutive images with FlowNet2 (Ilg et al., 2017). We then estimated the relative pose using stacked LSTM layers. A compact LSTM fusion model concatenates the absolute and relative poses and predicts optimized absolute poses. Our experiments were conducted on multiple datasets including Industry scenario #1, scenario #2, scenario #3, and Microsoft 7-Scenes (Shotton et al., 2013). The proposed approach yields a lower absolute pose pre-

Table 1.3: Summary of APR and RPR fusion techniques proposed in the contributing papers (Ott et al., 2020a, 2023a,b). The baseline models are PoseNet (Kendall et al., 2015), FlowNet2 (Ilg et al., 2017), IMUNet (do Monte Lima et al., 2019), FlowNet (Dosovitskiy et al., 2015), and Lucas-Kanade (Baker & Matthews, 2004). [s] indicates synthetic pre-training from simulation.

Model 1	Model 2	Fusion Technique
APRV: PoseNet	RPRV: FlowNet2	Pose concatenation + stacked LSTM layers
APRV: PoseNet	—	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV: PoseNet	RPR _I : IMUNet	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV: PoseNet	RPR _I : IMUNet	Layer concatenation
APRV: PoseNet	RPR _I : IMUNet	Layer concatenation + BiLSTM
APRV: PoseNet	RPR _I : IMUNet	Soft fusion (SSF) (Chen et al., 2019)
APRV: PoseNet	RPR _I : IMUNet	Soft fusion (SSF) (Chen et al., 2019) + BiLSTM
APRV: PoseNet	RPR _I : IMUNet	Multi-modal transfer module (Joze et al., 2020)
APRV: PoseNet	RPR _I : IMUNet	Auxiliary network (Navon et al., 2021) (non-linear)
APRV: PoseNet	RPR _I : IMUNet	Auxiliary network (Navon et al., 2021) (convolutional)
APRV: PoseNet	RPR _I : IMUNet	Bayesian network (BNN) (Kendall & Gal, 2017)
RPRV: FlowNet	RPR _I : IMUNet	Layer concatenation
RPRV: FlowNet	RPR _I : IMUNet	Layer concatenation + BiLSTM
RPRV: FlowNet	RPR _I : IMUNet	Soft fusion (SSF) (Chen et al., 2019)
RPRV: FlowNet	RPR _I : IMUNet	Soft fusion (SSF) (Chen et al., 2019) + BiLSTM
RPRV: FlowNet	RPR _I : IMUNet	Multi-modal transfer module (Joze et al., 2020)
APRV: SfM	RPRV: Lucas-Kanade	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV: SfM	RPRV: Lucas-Kanade	Eight different RNN fusion modules
APRV: SfM	RPRV[s]: Lucas-Kanade	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV: SfM	RPRV[s]: Lucas-Kanade	Eight different RNN fusion modules
APRV: PoseNet	RPRV: Lucas-Kanade	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV: PoseNet	RPRV: Lucas-Kanade	Eight different RNN fusion modules
APRV: PoseNet	RPRV[s]: Lucas-Kanade	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV: PoseNet	RPRV[s]: Lucas-Kanade	Eight different RNN fusion modules
APRV[s]: PoseNet	RPRV: Lucas-Kanade	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV[s]: PoseNet	RPRV: Lucas-Kanade	Eight different RNN fusion modules
APRV[s]: PoseNet	RPRV[s]: Lucas-Kanade	Pose graph optimization (PGO) (Grimes et al., 2010)
APRV[s]: PoseNet	RPRV[s]: Lucas-Kanade	Eight different RNN fusion modules

dition error compared to using APRV only, while also smoothing the global trajectory. This improvement is particularly noteworthy in scenarios where individual images contain either no or very few features (e.g., when the camera is directed towards a large white wall or black absorber wall) while the relative model can still generate a reliable relative pose from these limited features.

While the previous method employs only one modality (visual), our recent work (Ott et al., 2023b) leverages an additional modality, namely inertial data. This can prove advantageous in real-world applications where a camera failure is possible, and the model can rely on the inertial modality. Here again, we use PoseNet (Kendall et al., 2015) as the APRV model and employ IMUNet (do Monte Lima et al., 2019) as the RPR_I model. Regarding

the APR_V - RPR_I fusion, we evaluate six well-established techniques. Pose graph optimization (PGO) (Grimes et al., 2010) is a non-convex minimization problem that refines and improves the predicted absolute poses during inference by ensuring consistency between the predicted and refined relative poses. Fusion of late-layer features is a widely used technique. However, the straightforward concatenation of absolute and relative features does not allow the model to establish a relationship between them. By incorporating BiLSTM layers, the model can learn the connection between the current and previously predicted absolute poses, leading to improved pose prediction results. Additionally, we examined the soft fusion approach as a module of the selective sensor fusion (SSF) (Chen et al., 2019), originally developed for RPR. Both layer concatenation and soft fusion (with BiLSTM layers) perform similarly, with variations in pose error depending on the environment. The multi-modal transfer module (MMTM), introduced by Joze et al. (2020), enables the fusion of modalities with different spatial dimensions, and hence, we utilize this module as an intermediate fusion technique. We evaluate MMTM modules with one, two, or three layers between APR_V and RPR_I . We demonstrated that MMTM, originally developed for fusing high-dimensional image and video modalities, can also be applied to the pose estimation task by employing time-distributed models. Specifically, we fed three stacked consecutive images to the APR model and four stacked IMU samples from consecutive time steps to the RPR model. In addition, we evaluated the use of the auxiliary learning network (AuxiLearn), proposed by Navon et al. (2021), to treat the RPR model as an auxiliary task. However, a disadvantage is that the fusion model only predicts absolute pose, while the relative pose can still be relevant for transfer learning to new scenarios. The aleatoric uncertainty of poses can be measured to detect challenging images and assign different weights to the absolute and relative poses during inference. In our study, we evaluated the Bayesian neural network proposed by Kendall & Gal (2017). We observed an increase in aleatoric uncertainty and absolute pose prediction error for images with reflective surfaces. In addition to the APR_V - RPR_I fusion, we also investigated the fusion of two RPR modalities, namely RPR_V - RPR_I , to predict the relative pose. This is a sub-field of visual-inertial odometry, which estimates the motion of an object in 3D space. This technique is particularly useful in scenarios where GPS signals are not available or where training datasets are not available for the entire environment. In this study, we utilized FlowNet (Dosovitskiy et al., 2015) as the baseline model for RPR_V and IMUNet (do Monte Lima et al., 2019) as the baseline model for RPR_I . We evaluated two fusion techniques: late fusion (layer concatenation and soft fusion) and intermediate fusion using MMTM modules. Our experiments were conducted on multiple datasets including Inudstry scenario #4, EuRoC MAV (Burri et al., 2016), and PennCOSYVIO (Pfrommer et al., 2017). The MMTM fusion technique demonstrated the best performance with the lowest RPR prediction error, particularly when using three MMTM modules as intermediate fusion.

In the contributing paper (Ott et al., 2023a), a novel approach is presented to combine absolute poses (APR_V) with relative poses (RPR_V) derived from visual-only inputs. For the APR_V model, we employ a structure from motion (SfM) technique (Resch et al., 2015; Jiang et al., 2020b) to reconstruct a point cloud of the environment or a time-distributed PoseNet (Kendall et al., 2015). Relative poses are obtained by computing optical flow using

Lucas-Kanade (Baker & Matthews, 2004), and a small recurrent network with LSTM are utilized to predict poses. Similar to Ott et al. (2023b), to improve the absolute poses, we refine them with the relative poses using the state-of-the-art PGO algorithm. We propose a framework consisting of eight different recurrent fusion cells (Hochreiter & Schmidhuber, 1997; Chung et al., 2014; Zhou et al., 2016; Lee et al., 2017; Lei et al., 2017; Bradbury et al., 2017; Balduzzi & Ghifary, 2017; Laurent & von Brecht, 2017). We conducted experiments on all datasets of Industry scenario #4 with cross-validation. The proposed approach outperforms PGO with smoothed trajectories with fewer outliers. Notably, strongly-typed RNN (TRNN), introduced by Balduzzi & Ghifary (2017), significantly outperforms PGO and other RNN cells. In addition, the generalizability of the models is improved by pre-training both APR_V and RPR_V models with synthetic data from a simulated framework.

1.3 Deep Learning Foundations

The reader’s comprehension of this thesis necessitates a proficient understanding of deep learning, specifically with respect to convolutional neural networks (CNNs) employed for image-based classification (see Section 1.3.1), and spatio-temporal networks that assimilate data over time (see Section 1.3.2). The time-series classification task and associated architectures are scrutinized in Section 1.3.3. Additionally, Section 1.3.4 expounds on feature embeddings, which established the foundation for representation learning.

1.3.1 Neural Networks

Artificial neural networks draw inspiration from the biological processes of brain cells. A feedforward neural network is a directed acyclic graph composed of an input layer, one or more hidden layers, and an output layer. The network’s depth is determined by the number of layers, while the number of each layer dictates its width. Cells within a layer receive input signals from previous layer’s cells. Upon receiving the input signals, these cells produce a signal if the weighted sum of the input signal exceeds a predefined threshold. (Goodfellow et al., 2016; Ott, 2019) A neuron can be described through the equation

$$f(\mathbf{x}) = \sigma\left(\sum_{i=1}^n w_i x_i + b\right) = \sigma(\mathbf{w}^T \mathbf{x} + b), \quad (1.5)$$

where n is the number of input values, \mathbf{w} is a weight vector, \mathbf{x} is the input vector, b is the bias, and σ is a non-linear activation function. The standard activation functions commonly used are the Sigmoid function, the logistical function, the hyperbolic tangent function, and a rectified linear unit (ReLU) function (Goodfellow et al., 2016). In contrast to feedforward neural networks, CNNs accept three-dimensional input, such as an image, and are defined by their convolution operation. CNNs are comprised of three distinct types of layers: convolutional, pooling, and fully connected (FC) layers. A convolutional layer applies a filter to the input pixels, multiplied with a convolution kernel. Pooling layers decrease the feature map’s resolution and acquire spatial invariance. In general, a network

culminates with FC layers that accept an input volume and output a K -dimensional vector, where K represents the number of classes in the classification task. For the OnHW recognition task (which classifies 52 characters), the FC layer yields a 52-dimensional vector. The neural network is trained by minimizing an error through backpropagation of a cost/loss function $\mathcal{L}(\mathbf{w})$. This implies that the neurons' weights are updated in the direction of steepest descent in E through $w_i \leftarrow w_i - \eta \frac{\partial E}{\partial w_i}$, with a learning rate of η . In our OnHW recognition application, we employ the Adaptive movement estimation (Adam) optimizer (Kingma & Ba, 2014). Additionally, stochastic gradient descent (SGD) is commonly utilized. (Goodfellow et al., 2016; Ott, 2019)

1.3.2 Temporal Networks

Over the past decade, temporal networks have received considerable attention in contrast to standard convolutional networks due to their advantageous properties for applications that include spatio-temporal data. Examples of these applications are online handwriting recognition (Ott et al., 2022d), human activity recognition (Anguita et al., 2013; Stisen et al., 2015), motion detection (Blankertz et al., 2001), time-series forecasting, and radio frequency-based localization (Stahlke et al., 2022). This section provides a brief overview of temporal networks, such as recurrent neural networks (RNNs), long short-term memories (LSTMs), bidirectional LSTMs (BiLSTMs), and temporal convolutional networks (TCNs), which lay the foundation for the methods discussed in Section 1.3.3.

RNNs. An RNN is a feedforward neural network extension that includes connections between nodes that form a directed graph. RNNs can represent time sequences with dynamic temporal behavior. Given the input value \mathbf{x} of length t , the state \mathbf{s}_t of the classical form is computed by $\mathbf{s}_t = f(\mathbf{s}_{t-1}; \theta)$, where θ represents the network parameters. By unrolling this loop, each node \mathbf{s}_t in an RNN is related to the node \mathbf{s}_{t+1} in a directed acyclic graph. Thus, the state \mathbf{s}_t contains information about all previous states $\mathbf{s}_0, \dots, \mathbf{s}_{t-1}$. The computation of states in an RNN is given by $\mathbf{s}_t = \sigma(\mathbf{U}\mathbf{x}_t + \mathbf{W}\mathbf{s}_{t-1} + \mathbf{b}_t)$, where \mathbf{h}_t represents the output of node \mathbf{s}_t , and \mathbf{b}_t denotes the bias. The sigmoid function is represented by σ . The RNN output is $\mathbf{h}_t = \text{softmax}(\mathbf{V}\mathbf{s}_t)$, where \mathbf{U} , \mathbf{W} , and \mathbf{V} denote the hidden layer matrices (Goodfellow et al., 2016). Due to the reconstruction of a vector of activations for each time step, RNNs are quite deep and may suffer from the issue of vanishing and exploding gradients. However, despite their difficulty to train, RNNs can still be beneficial for small-scale problems (Ott, 2019). Numerous variants of RNNs have been proposed in the literature to address the issue of vanishing and exploding gradients (Hochreiter & Schmidhuber, 1997; Chung et al., 2014; Zhou et al., 2016; Lee et al., 2017; Lei et al., 2017; Bradbury et al., 2017; Balduzzi & Ghifary, 2017; Laurent & von Brecht, 2017). Among them, the LSTM cell is the most widely used and is described below.

LSTMs. LSTMs (Hochreiter & Schmidhuber, 1997) solve the problem of vanishing gradients by incorporating an input gate unit that prevents the perturbation of memory

contexts. In addition, they also include an output gate and a memory unit. Unlike RNNs, LSTMs consist of a chain of repeating cells instead of a single layer (Ott, 2019).

Bidirectional Units. BiLSTMs (Graves et al., 2009) are a variant of LSTMs, in which the input information flows in both forward and backward directions. This is achieved by adding an extra LSTM layer that operates in reverse order. BiLSTMs leverage information from both past and future contexts, which is crucial for capturing sequential dependencies, such as those between words and phrases. The advantage of BiLSTMs is that each component, such as each character of a word, has access to both past and future information. However, the training time of BiLSTMs is significantly higher than that of LSTMs.

TCNs. To encode spatio-temporal information and capture high-level temporal information, TCNs (Bai et al., 2018) utilize a combination of CNNs and classifier networks. The CNN is used to compute low-level features, which are then passed to the classifier network. The encoder-decoder network is capable of processing input sequences of arbitrary length and generating output sequences of equal length. To ensure that information is not lost from future to past, TCNs utilize causal convolutions, which convolve the output only with elements from the same time or earlier.

1.3.3 Time-Series Classification & Architectures

In this section, we present a mathematical definition of the time-series classification task and provide an overview of the state-of-the-art time-series classification architectures that we employ in our experiments.

Time-Series Classification Task. In Section 1.2.1, we defined the classification task for single characters and sequences of characters for the OnHW recognition task. In the following, we repeat and provide a general mathematical definition of multivariate time-series and the classification task (Abanda et al., 2022). We use this definition in the contributing papers (Ott et al., 2020b, 2022a,b,c,d; Klaß et al., 2022; Ott et al., 2023c) and in Chapter 2.

Definition 1.3.1 (Multivariate Time-Series). *A multivariate time-series (MTS) $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $l \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l})$, $i \in \{1, \dots, m\}$, where $m \in \mathbb{N}$ is the length of the time-series.*

Definition 1.3.2 (MTS Classification Task). *The MTS training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\} \in \mathbb{R}^{n_U \times m \times l}$, where n_U is the number of time-series. Each MTS is associated with v from a pre-defined label set Ω with K classes. The aim of MTS classification is to predict an unknown class label $v \in \Omega$ for a given MTS. The labels $\mathcal{V} = \{v_1, \dots, v_{n_U}\} \in \Omega^{n_U}$ are the corresponding training labels to the training MTS set \mathcal{U} .*

FCN. The time-series classification task was addressed by Wang et al. (2016) using fully convolutional networks (FCNs), which extracts features and use a global pooling layer with softmax activation as a classifier to predict classes. Each FCN block comprises a convolutional layer with a batch normalization layer and ReLU activation. The FCN model consists of three such basic blocks. The compact size of the model enables FCN to achieve high accuracy and require a short training time when applied to small datasets.

RNN & LSTM & GRU & BiLSTM. Recurrent networks, including RNNs, LSTMs, and GRUs (see Section 1.3.2), have become the predominant choice for sequence modeling (Chung et al., 2014). BiLSTMs (Graves et al., 2009) have been specifically designed for data that is difficult to segment and contains long-range bidirectional interdependencies. Among recurrent units, BiLSTMs have outperformed others, especially in the case of sequence-to-sequence classification in the OnHW setup for long sequences (Ott et al., 2022d).

TCN. TCNs, introduced by Bai et al. (2018), are simple convolutional architectures that can outperform RNNs on a diverse range of tasks. For more information, see Section 1.3.2. We evaluated a CNN in combination with TCNs on the OnHW recognition task in our experiments (Ott et al., 2022d).

(M)RNN-FCN & (M)LSTM-FCN & (M)GRU-FCN. Many techniques combine convolutional networks, such as CNNs and FCNs, with temporal layers including RNNs, LSTMs, GRUs, or TCNs. These are LSTM-FCN (Karim et al., 2017) and GRU-FCN (El-sayed et al., 2019). Multi-dimensional networks, such as the multivariate LSTM (MLSTM) by Karim et al. (2019), scan the input in all four possible directions, and compute the RNN inner states and output based on the previous position in the vertical and horizontal directions. The advantage of MRNN-FCN and their variants is that they can capture complex dependencies in multiple dimensions, such as spatial and temporal dimensions. This is particularly useful for multi-modal time-series data. However, due to their increased complexity, multi-dimensional networks have not been commonly used in recent years.

ResCNN. The ResCNN (Zou et al., 2019) is a hybrid method that incorporates residual networks into CNNs by introducing them into the first three convolutional layers. By using different activation functions in different layers, the model’s performance can be improved.

ResNet & XResNet. ResNet (Wang et al., 2016) is an extension of FCN that employs shortcut connections in each residual block, allowing for deeper networks and gradient flow through the bottom layers. Three residual blocks are stacked, each with three filters. However, ResNet is prone to overfitting on small datasets. XResNet (He et al., 2019) improves parallelism during training and reduces computational cost by modifying convolutional layers.

InceptionTime & XceptionTime. InceptionTime (Fawaz et al., 2020), which is an ensemble of five deep CNN models inspired by the Inception-v4 model (Szegedy et al., 2017), consists of two residual blocks with each block containing three Inception modules. The output MTS is averaged over the time dimension using a global average pooling layer after the residual blocks. An Inception module first applies a bottleneck layer (a sliding filter) that reduces the MTS dimensionality, followed by multiple sliding filters of varying lengths to the output of the bottleneck layer. Our implementation of InceptionTime, combined with BiLSTM layers, performed well on the OnHW classification tasks as reported in Ott et al. (2022b,d, 2023c). On the other hand, XceptionTime (Rahimian et al., 2019) uses a series of XceptionTime modules with residual connections, followed by 1D convolutional, batch normalization, and average pooling layers for the classification task. The XceptionTime modules consist of depthwise separable convolutions and max pooling layers in two parallel paths.

Transformer Models. Recently, Transformer models have gained popularity for visual recognition tasks, but they have also been used for MTS classification. For instance, Zerveas et al. (2021) developed an unsupervised time-series transformer (TST) that uses Gaussian noise corruption. While TST performs comparably with the CNN+BiLSTM model on the OnHW classification tasks, the adaptation of visual transformer models, such as Perceiver (Jaegle et al., 2021), Sinkhorn (Tay et al., 2020), Performer (Choromanski et al., 2021), Reformer (Kitaev et al., 2020), and Linformer (Wang et al., 2020), is not intuitive. Therefore, the performance of these models drops on the OnHW classification tasks due to the requirement of large amounts of training data.

TapNet. TapNet, a prototypical network proposed by Zhang et al. (2020b), learns low-dimensional features from time-series data by using a combination of 1D convolutional and LSTM layers and designing a random group permutation method. The model then computes probabilities over classes based on the class prototypes and the time-series data by calculating distances between low-dimensional representations using Bregman divergences (refer to Definition 1.6.3). In the contributing paper (Ott et al., 2022c), we presented a visualization of the class clusters and their prototypes computed by TapNet in a lower-dimensional space.

XEM. The explainable-by-design ensemble method for MTS classification, proposed by Fauvel et al. (2022), combines a boosting-bagging approach with a divide-and-conquer approach. The method, known as XEM, aims to achieve a balance between model performance and faithful explainability.

1.3.4 Feature Embeddings

The goal of feature extraction is to learn a low-dimensional representation, referred to as feature embeddings, for each instance in order to preserve the information contained in

its features. These feature embeddings act as hidden representations for a given object. Given an MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$, as introduced in Definition 1.3.1, the feature embedding of the MTS is defined as follows.

Definition 1.3.3 (Feature Embedding). *A feature embedding of a neural network is defined by $f(\mathbf{U}) \in \mathbb{R}^{q \times w}$ for the corresponding MTS \mathbf{U} . $q \times w$ is the size of the output of the neural network layer.*

A feature embedding is typically the output of an activation function, such as a sigmoid or ReLU activation, of a neural network layer. This definition can be extended to three-dimensional feature embeddings, as in the case of time-distributed image-based classification tasks. Feature embeddings are utilized in the multi-modal learning task (discussed in Section 1.4), where they are shared between networks of different modalities, and in the domain adaptation task (discussed in Section 1.5), where the discrepancy between feature embeddings is minimized between different domains.

1.4 Cross-Modal Retrieval & Multi-Modal Learning

This section presents the concept of cross-modal retrieval (CMR), which aims to achieve a shared task or common representation across diverse data sources. The distinct modalities are examined in Section 1.4.1. Furthermore, we provide a summary of CMR techniques and explain the different types of fusion in Section 1.4.2. Additionally, to improve the learning process, negative entities can be employed in the CMR setup, which is known as pairwise or contrastive learning. This technique is elaborated in Section 1.4.3.

1.4.1 Modalities

In the context of ML, the term *modality* pertains to the category of input data or information that a model is capable of processing, and may originate from various data sources. One of these sources are IMU sensors that commonly exist in smartphones, tablets, and smartwatches, which includes accelerometers, gyroscopes, and magnetometers. The modality of the data is presented as an MTS and can be utilized for recognizing human activities. Another modality is obtained from signals captured by electroencephalography (EEG) devices, which can be used for classifying sleep stages. The audio modality is derived from time-series data recorded through microphones. In computer vision, modality refers to the category of visual data, such as images or videos that are captured using cameras. Various image modalities are available, including monocular, stereographic, or RGB-D images. In the context of natural language processing, modality is linked to the category of textual data, which can be in written or spoken form. Trajectory modalities refer to two-dimensional time-series data that contains information about the x - and y -coordinates of objects, which is utilized in various applications, such as robotics, sports analysis, and epidemiology. It is worth mentioning that in numerous applications, data inputs of the same modality but obtained from diverse input devices are utilized to encompass a wider range

of information. To illustrate, in the domain of human activity recognition, accelerometers are attached to both the hand and the leg to capture a wider range of movement data. Recognizing the data modality is crucial in developing appropriate preprocessing techniques, feature extraction methods, and ML architectures. Since each modality comprises distinct information pertaining to the objective, utilizing two or more modalities concurrently to train the ML model can enhance the outcome. As the dimensionality of modalities can vary significantly, for instance, visual versus time-series data, employing an advanced CMR technique is necessary to optimally extract and merge modalities (see Section 1.4.2).

Application in Contributing Papers. In our contributing papers, we employ diverse types of modalities for different applications. For OnHW recognition, we usually use sensor streaming data obtained from IMUs as model input (Ott et al., 2020b, 2022a,b,c,d; Klaß et al., 2022; Ott et al., 2023c). In Ott et al. (2022b), we simultaneously learn a common representation between MTS data recorded on paper and on a tablet. Although the data source and type of both modalities are identical, the noise level differs between the two MTS data. In Ott et al. (2023c), we improve the main task by introducing generated visual data. Apart from IMU data for OnHW recognition, we also utilize video recordings to capture the outside-in trajectory coordinates of the pen tip (Ott et al., 2022c). In the context of self-localization, the main modality is monocular images. In Ott et al. (2020a, 2023a), we augment the principal visual self-localization task by incorporating an additional modality, specifically optical flow presented as image modalities. While we use the same principal visual modality in Ott et al. (2023b), we enhance the task by incorporating an auxiliary inertial modality acquired from IMU measurements.

1.4.2 Methods & Fusion Points

CMR Methods. The concept of multi-modal learning involves using data from multiple modalities to train a model to perform a specific task, which can lead to better performance than using a single modality. However, this approach poses challenges such as data alignment, modality selection, and model fusion. In contrast, CMR retrieves information of one modality using a sample from a different modality. In our paper (Ott et al., 2023c), we provide an overview of CMR methods. As research for multi-modal learning methods are wide-ranged and there exist a lot of methods, we focus on the general concept of CMR and our utilized methods. The multi-modal transfer module (MMTM) (Joze et al., 2020) allows for fusion of modalities with different spatial dimensions by pooling spatial information into channel descriptors. The modality mixer (M-Mixer) (Lee et al., 2023) introduces a recurrent unit, the multi-modal contextualization unit (MCU), to encode a sequence of one modality with features of other modalities at intermediate layers. The late fusion approach, namely soft fusion (Chen et al., 2019), employs an attention mechanism by element-wise multiplying sigmoid masks to the features.

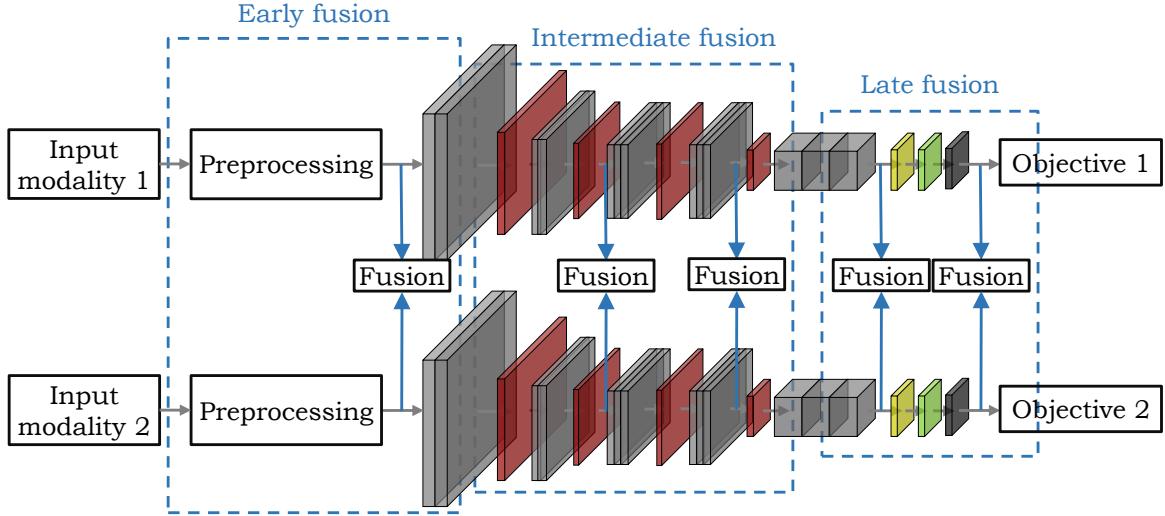


Figure 1.11: Overview of a multi-modal learning fusion architecture. Two (or more) input modalities are processed separately, features are extracted with a neural network, and the objective (at least one) is minimized. The features between both networks are shared with early, intermediate, or late fusion modules. The colored blocks indicate different layers: one- or two-dimensional convolutional layers (grey), pooling layers (red), spatio-temporal layers such as LSTM, TCN, or BiLSTM layers (yellow), dropout (green), and dense layers (black).

Fusion Points. Figure 1.11 illustrates a network architecture that fuses two distinct modalities. Depending on the fusion level, three different types of fusion are commonly used: (1) Early fusion combines the raw modalities either before or after preprocessing, but this approach is poorly explored in the literature. Another early fusion approach is to combine modalities in their lower-dimensional space using correlation analysis. (2) Intermediate fusion combines features from all intermediate layers, which are diverse but have a weak expressive capacity, resulting in noisy features that can cause confusion between similar classes. (3) Lastly, late fusion combines modality-wise objective features by extracting decisions from single-modality architectures and then fusing them for a final decision. These methods often recover scores from the softmax layers. (Boulahia et al., 2021)

Application in Contributing Papers. The fusion process in the architecture varies depending on the application. In the case of the DA task involving time-series data with domain shift (Ott et al., 2022a), we utilized a small CNN+BiLSTM model. We perform a feature embedding transformation between two models that are essentially the same. Specifically, we select a late intermediate layer, which refers to the output of the last convolutional layer before the two BiLSTMs. This particular layer contains all the necessary information for the DA task, and its layer size is advantageous for optimal transport techniques that scale with the input size. In the context of OnHW recognition, the layer size is 19×200 , while for the sinusoidal classification task, the layer size is 50×30 . We learn a

common representation between offline and online HWR using intermediate layers to enhance the online HWR task (Ott et al., 2023c). Specifically, two intermediate convolutional layers after the visual and time-series encoders are chosen. For the multi-modal learning setup, the use of softmax activation is crucial. In contrast, for the single task classification, ReLU activation is preferred. The tablet and paper handwriting recognition tasks are improved using contrastive learning by fusing two CNN+BiLSTM layers (Ott et al., 2022b). In this study, we assess five intermediate and late points for DA after applying batch normalization, BiLSTM, and dense layers. While intermediate DA consistently reduces the CER due to more trainable parameters after fusion, late DA is reliant on the representation loss. Furthermore, in Ott et al. (2022c), we evaluate eight different split points. In contrast to fusion, this model only takes in one data modality as input but concurrently and jointly solves a classification and regression task. Therefore, we refer to split points, where features are processed independently, instead of fusion points. This is essential because the trajectory regression task is more difficult than the character classification task, requiring more trainable parameters at the regression trunk. From our experiments, we observe that a late split has a positive impact on the trajectory regression task. In another research paper, we introduce ViPR (Ott et al., 2020a), which utilizes concatenation to combine absolute and relative pose estimations, represented by the latest fusion point. In Ott et al. (2023b), we perform feature fusion of layers that contain information regarding absolute and relative poses. We evaluate three fusion techniques: concatenation (late fusion), soft fusion (Chen et al., 2019) (late fusion), and intermediate fusion with MMTM (Joze et al., 2020). Adding BiLSTM layers is a critical step in extracting necessary spatio-temporal pose information. In Brieger et al. (2022), we explore different fusion techniques between ResNet18 and time-series Transformer (TS-Transformer) for interference detection (van der Merwe et al., 2023) in global navigation satellite system (GNSS) signals using visual and time-series data. In our evaluation, we consider three fusion techniques: late fusion, which involves layer concatenation and soft fusion (Chen et al., 2019), and intermediate fusion with MMTM (Joze et al., 2020). Our results show that the highest accuracy is attained using concatenation, with a 95% F- β score, followed by soft fusion with 93%, and MMTM with 89%. This approach improves the baseline accuracy of TS-Transformer (59%) and ResNet18 (88%) models.

1.4.3 Contrastive & Triplet Learning

Many applications suffer from small training datasets due to the time-consuming nature of data recording. Consequently, ML models tend to overfit on the training data. To address this issue, recent methods utilize negative identities of each sample to enhance the dataset. One such technique is *contrastive* or *triplet* learning, which we discuss in the following section.

Contrastive Learning. We briefly define contrastive learning, as previously introduced in Ott et al. (2023c). In joint representation learning, it is typical to minimize the distance between embeddings for samples of the same class (see Section 1.6). The ML model is

trained such that samples of the same class have a small embedding distance, while samples of distinct classes have a large distance. Recent methods have extended this approach by minimizing the distance between the embeddings of the anchor sample \mathbf{U}^a and the positive sample \mathbf{U}^p (both samples have the same class label) and maximizing the distance between the embeddings of the anchor sample \mathbf{U}^a and the negative sample \mathbf{U}^n (both samples have different class labels). The feature embedding $f(\mathbf{U})$ of the sample \mathbf{U} is defined in Section 1.3.4. The objective is to minimize the loss function of an ML model such that the following inequality is satisfied for all training samples $(f(\mathbf{U}_i^a), f(\mathbf{U}_i^p), f(\mathbf{U}_i^n)) \in \Theta$:

$$\mathcal{L}_c(f(\mathbf{U}_i^a), f(\mathbf{U}_i^p)) + \gamma < \mathcal{L}_c(f(\mathbf{U}_i^a), f(\mathbf{U}_i^n)). \quad (1.6)$$

Θ is a set of all possible triplets, $\mathcal{L}_c(\cdot)$ is a metric loss as defined in Section 1.6, and γ is a margin between positive and negative samples, as described by Schroff et al. (2015). The contrastive loss function considers the distance between the embeddings of the anchor and the positive identity *separately* from the distance between the embeddings of the anchor and the negative identity. Specifically, we define \mathbf{U}_i^a , \mathbf{U}_i^p , and \mathbf{U}_i^n as data samples from the same modality. However, the inequality can be also formulated to learn between different modalities, where \mathbf{U}_i^a is from one (the main) modality, and \mathbf{U}_i^p and \mathbf{U}_i^n are from another (the auxiliary) modality (Ott et al., 2023c).

Triplet Learning. The triplet loss function differs from the contrastive loss function in that it minimizes the distance between the embeddings of the anchor and the positive identity, while *simultaneously* maximizing the distance to the embeddings of the negative identity. This is achieved using the following loss function:

$$\mathcal{L}_t(f(\mathbf{U}^a), f(\mathbf{U}^p), f(\mathbf{U}^n)) = \sum_{i=1}^N \max \left[\mathcal{L}_c(f(\mathbf{U}_i^a), f(\mathbf{U}_i^p)) - \mathcal{L}_c(f(\mathbf{U}_i^a), f(\mathbf{U}_i^n)) + \gamma, 0 \right], \quad (1.7)$$

where N is the number of triplets in Θ (Schroff et al., 2015; Ott et al., 2023c). $\mathcal{L}_c(\cdot)$ is a deep metric loss function as defined in Section 1.6.

Application in Contributing Papers. The selection of triplet pairs for training is a challenging task as certain negative samples, such as soft and hard samples, can reduce the model’s performance. Therefore, the goal is to select semi-hard samples (Do et al., 2019). While the triplet loss is typically used for a single-class classification task, we advocate using the triplet loss for sequence-based learning, specifically for classifying words. In Ott et al. (2022b, 2023c), we propose selecting negative samples based on a dynamic margin that is predicated on the Edit distance (see Equation 1.2). Using the dynamic margin approach, hard negative samples are selected during the early stage of training, while semi-hard negative samples are selected during the later training stages. In our work, we enhanced the main sequence-to-sequence OnHW recognition task for writing on paper with negative samples for writing on tablet devices (Ott et al., 2022b). The negative samples are selected using the dynamic margin. Additionally, in Ott et al. (2023c), we applied the same

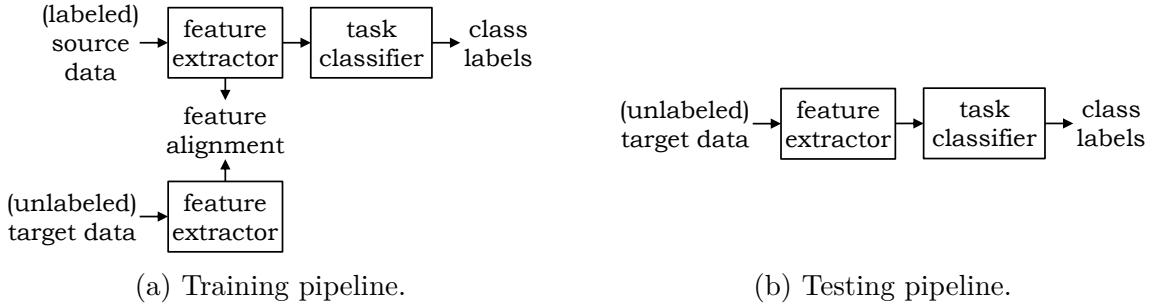


Figure 1.12: General pipeline for DA methods that learn a domain-invariant representation between (labeled) source data and (unlabeled) target data (Wilson & Cook, 2020).

dynamic margin between an inertial MTS modality (online HWR) and a visual modality (offline HWR). The present approach demonstrates an enhanced accuracy in classification and faster rate of convergence. Additionally, the model exhibits generalization capabilities across distinct writers (Ott et al., 2023c) and diverse writing surfaces (Ott et al., 2022b). In contrast, the latest comparable technique proposed by Liu et al. (2022) applies a contrastive loss for recommending among three modalities, namely images, description text, and item graph. The description text and image modalities work in synergy, through a task of inter-modal alignment, by contrasting each other for the same item.

1.5 Domain Adaptation

The prevalence of deep learning in various classification tasks is largely dependent on the availability of large labeled datasets. However, annotating these datasets can be challenging and expensive, requiring expert knowledge for many applications. To overcome this challenge, one approach is to train the model with data from a source domain and apply the model to the target domain – the domain of interest. Figure 1.12 presents the general pipeline for domain adaptation (DA) methods. Since the data from the source and target domains are from different distributions, there is a domain shift between the two domains, which leads to reduced model performance on the target domain. DA methods, which are a sub-category of transfer learning methods, are employed to reduce this domain shift. DA techniques are necessary in applications ranging from human activity recognition (Liu et al., 2009; Kwapisz et al., 2010; Anguita et al., 2013; Stisen et al., 2015), online handwriting recognition (Alimoglu & Alpaydin, 1997; Ott et al., 2022a; Klaß et al., 2022), sleep stage classification (Goldberger et al., 2000), medical applications (Villar et al., 2016), or interference detection (Raichur et al., 2022; Brieger et al., 2022; Goswami et al., 2023; van der Merwe et al., 2023). In this section, we will delve into the field of DA.

Section 1.5.1 provides an explanation of the domain shift and the objective of DA. The mathematical boundary for DA is defined in Section 1.5.2. We present an overview of DA methods in Section 1.5.3, while Section 1.5.4 delves into optimal transport methods in more detail.

1.5.1 Definitions & Notations

Definition of Domain Shift & Domain Adaptation. First, we provide definitions for two important terms: *domain* and *task*. A domain comprises a feature space and a marginal probability distribution, while a task comprises a label space and an objective predictive function. According to this definition, changes in a domain may arise from alterations in either the feature space or the marginal probability, and the same holds for changes in a task (Wilson & Cook, 2020). In machine learning models, it is assumed that training and test datasets are independent and identically distributed (i.i.d.). However, the performance of a machine learning model deteriorates when it is applied to data from a domain that is similar but different (i.e., *source domain*) from the data it was initially trained on (i.e., *target domain*). This phenomenon is known as *domain shift*, which is also frequently referred to as *dataset bias* or *distributional shift*, between the source and target domains. In practice, this assumption of i.i.d. data rarely holds as real-world data often varies over time and space (Sun et al., 2016). To address this domain shift problem, DA techniques transfer knowledge from the target to the source domain, enabling the learning of a domain-invariant representation that reduces the domain discrepancy.

Notations. A domain \mathcal{D} consists of a feature space \mathcal{U} with marginal probability $P(\mathcal{U})$. The task is defined by the label space \mathcal{Y} . The joint distribution is $P(\mathcal{U}, \mathcal{Y})$ and the conditional distribution is denoted as $P(\mathcal{Y}|\mathcal{U})$. When considering two domains with domain shift, there is a source domain $\mathcal{D}_S = \{\mathcal{U}_S^i, \mathcal{Y}_S^i\}_{i=1}^{n_S}$ of $n_S \in \mathbb{N}$ labeled samples of $|\mathcal{Y}_S^i| \in \mathbb{N}$ categories, and a target domain $\mathcal{D}_T = \{\mathcal{U}_T^i, \mathcal{Y}_T^i\}_{i=1}^{n_T}$ of $n_T \in \mathbb{N}$ labeled samples of $|\mathcal{Y}_T^i| \in \mathbb{N}$ categories. Due to the difference of the two domains, the distributions are assumed to be different: $P(\mathcal{U}_S) \neq P(\mathcal{U}_T)$ and $P(\mathcal{Y}_S|\mathcal{U}_S) \neq P(\mathcal{Y}_T|\mathcal{U}_T)$ (Zhang, 2021; Ott et al., 2022a).

1.5.2 Bound for Domain Adaptation

In order to mitigate the domain shift between the source and target domains, a number of DA methods have been developed which constrain the target error by the sum of the source error and a distance measure between the two distributions. These methods operate under the assumption that the source risk provides a reliable estimate of the target risk when the two distributions are similar. Any distance measure discussed in Section 1.6 may be employed, though we will focus on the \mathcal{H} -divergence with dimension d . Specifically, \mathcal{H} is a set of binary classifiers $\eta : \mathcal{U} \rightarrow \{0, 1\}$, where \mathcal{U} is the label space as defined with Definition 1.3.1. Although, the same principles apply to multi-class settings (Ganin et al., 2016). In this section, we first introduce the \mathcal{H} -divergence in Definition 1.5.1 to summarize the target risk in Theorem 1.5.1, as introduced by Ganin et al. (2016).

Definition 1.5.1 (\mathcal{H} -divergence, Kifer et al. (2004); Ben-David et al. (2009); Ganin et al. (2016)). *Given the hypothesis space \mathcal{H} and two domain distributions $\mathcal{D}_S^{\mathcal{U}}$ and $\mathcal{D}_T^{\mathcal{U}}$ over \mathcal{U} , the \mathcal{H} -divergence between $\mathcal{D}_S^{\mathcal{U}}$ and $\mathcal{D}_T^{\mathcal{U}}$ is defined by*

$$d_{\mathcal{H}}(\mathcal{D}_S^{\mathcal{U}}, \mathcal{D}_T^{\mathcal{U}}) = 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{u \sim \mathcal{D}_S^{\mathcal{U}}}[\eta(u) = 1] - \Pr_{u \sim \mathcal{D}_T^{\mathcal{U}}}[\eta(u) = 1] \right|. \quad (1.8)$$

For a symmetric hypothesis class \mathcal{H} , the empirical \mathcal{H} -divergence can be computed between the samples $S \sim (\mathcal{D}_S^{\mathcal{U}})^{n_S}$ and $T \sim (\mathcal{D}_T^{\mathcal{U}})^{n_T}$ by

$$\hat{d}_{\mathcal{H}}(S, T) = 2 \left(1 - \min_{\eta \in \mathcal{H}} \left[\frac{1}{n_S} \sum_{i=1}^{n_S} I[\eta(x_i) = 0] + \frac{1}{n_T} \sum_{i=n_T+1}^{n_S+n_T} I[\eta(x_i) = 1] \right] \right), \quad (1.9)$$

where $I[\alpha]$ is 1 if α is true and 0 if α is false (Ganin et al., 2016). As it is hard to compute $\hat{d}_{\mathcal{H}}(S, T)$ exactly, by learning to discriminate between source and target samples, one can approximate $\hat{d}_{\mathcal{H}}(S, T)$ by the Proxy \mathcal{A} -distance $\hat{d}_{\mathcal{A}} = 2(1 - 2\epsilon)$, where ϵ is the generalization error. The \mathcal{A} -distance is defined by $d_{\mathcal{A}}(\mathcal{D}_S^{\mathcal{U}}, \mathcal{D}_T^{\mathcal{U}}) = 2 \sup_{A \in \mathcal{A}} |\Pr_{\mathcal{D}_S^{\mathcal{U}}}(A) - \Pr_{\mathcal{D}_T^{\mathcal{U}}}(A)|$, where \mathcal{A} is a subset of \mathcal{U} . Furthermore, Ben-David et al. (2009); Ganin et al. (2016) showed that $\hat{d}_{\mathcal{H}}(S, T)$ plus a constant complexity term is the upper bound to the \mathcal{H} -divergence $d_{\mathcal{H}}(\mathcal{D}_T^{\mathcal{U}}, \mathcal{D}_S^{\mathcal{U}})$. The ideal joint hypothesis minimizes the combined error $h^* = \arg \min_{\eta \in \mathcal{H}} R_{\mathcal{D}_S}(\eta) + R_{\mathcal{D}_T}(\eta)$. With this, the combined error is $\lambda = R_{\mathcal{D}_S}(h^*) + R_{\mathcal{D}_T}(h^*)$.

Theorem 1.5.1 (Target risk, Ben-David et al. (2009); Ganin et al. (2016)). *For any $\delta \in (0, 1)$, with the probability at least $1 - \delta$ over the choice of samples $S \sim (\mathcal{D}_S^{\mathcal{U}})^{n_S}$ and $T \sim (\mathcal{D}_T^{\mathcal{U}})^{n_T}$, it holds for every $\eta \in \mathcal{H}$:*

$$R_{\mathcal{D}_T}(\eta) \leq R_{\mathcal{D}_S}(\eta) + \hat{d}_{\mathcal{H}}(S, T) + 4 \sqrt{\frac{2d \log(2q) + \log(\frac{2}{\delta})}{q}} + \lambda, \quad (1.10)$$

with the size q of the samples $\mathcal{U}_S, \mathcal{U}_T$. $\lambda \geq \inf_{\eta^* \in \mathcal{H}} [R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*)]$, and the empirical source risk

$$R_{\mathcal{D}_S}(\eta) = \frac{1}{n} \sum_{i=1}^n I[\eta(x_i) \neq y_i]. \quad (1.11)$$

The goal is to find a representation of the samples, such that the source and the target domains are as indistinguishable as possible, and hence, a hypothesis with a low source risk performs well on the target data. For more information and proofs, see Ben-David et al. (2009); Wilson & Cook (2020). Concluding from this, in the contributing paper (Ott et al., 2022a), we map the features of the source domain samples on the feature means of the target domain samples (extracted from pre-trained models), while leaving the target features unchanged. This approach aims to learn a representation that makes the source and target domains indistinguishable. In Ott et al. (2022b, 2023c) we propose a technique for learning a common representation by adjusting the weights of both source and target networks to align the two domains. In the following section, we give a succinct summary of techniques aimed at minimizing the aforementioned objective risk.

1.5.3 Categorization of Domain Adaptation Methods

DA is a subcategory of transfer learning (TL) that involves using labeled data to learn an effective representation network. In recent years, self-supervised learning has emerged

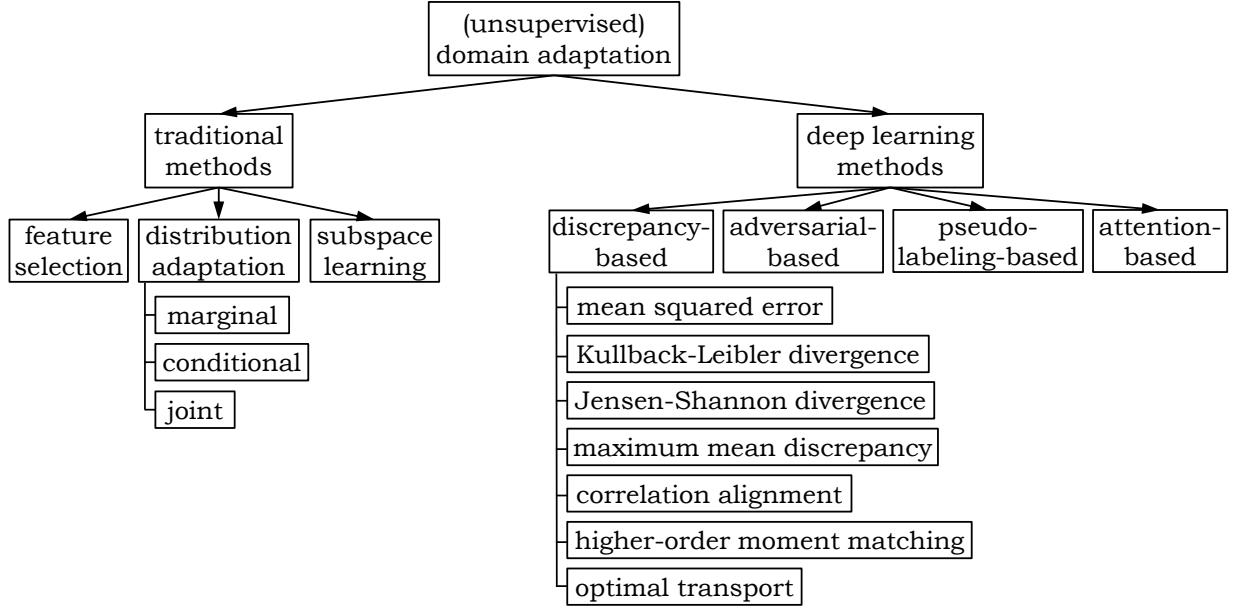


Figure 1.13: Overview of (unsupervised) DA categories and methods (Zhang, 2021). We focus on distribution adaptation and discrepancy-based methods.

as a popular approach for learning representations without the need for annotated labels. While recent research has focused on improving unsupervised learning for TL and DA, self-supervised learning has demonstrated superior performance in certain applications. Yang et al. (2020) reported that self-supervised learning outperforms TL when there is a significant domain difference or when the amount of pre-training data is limited. Additionally, self-supervised learning is more robust to class imbalance compared to TL, which is consistent with the findings of Ott et al. (2022a).

In the subsequent section, we categorize contemporary DA approaches. DA can be classified into three primary types: Firstly, *supervised* DA, where labeled data from the target domain is present during training. Secondly, *unsupervised* DA, which involves a labeled source domain and an unlabeled target domain. Finally, *semi-supervised* DA encompasses scenarios where a limited amount of labeled data from the target domain is available for training, along with unlabeled data. To overcome the constraints caused by insufficient annotations, techniques merge the labeled source domain with the unlabeled target domain. The majority of research are conducted in the domain of unsupervised DA. Generally, the number of class labels in the source and target domains are equivalent, known as closed set DA. Additionally, DA approaches can be categorized as either *homogeneous* or *heterogeneous*. Homogeneous techniques employ the same feature space with the same feature dimensionality, while heterogeneous methods employ different feature spaces and feature dimensionality (Zhang, 2021). The majority of DA approaches tackle the homogeneous DA problem. We alleviate the constraint (i.e., input data) of the DA methodologies' application, as they can typically be adapted to alternative data sources, ranging from visual data to time-series data and vice versa. In the subsequent categorization, we emphasize

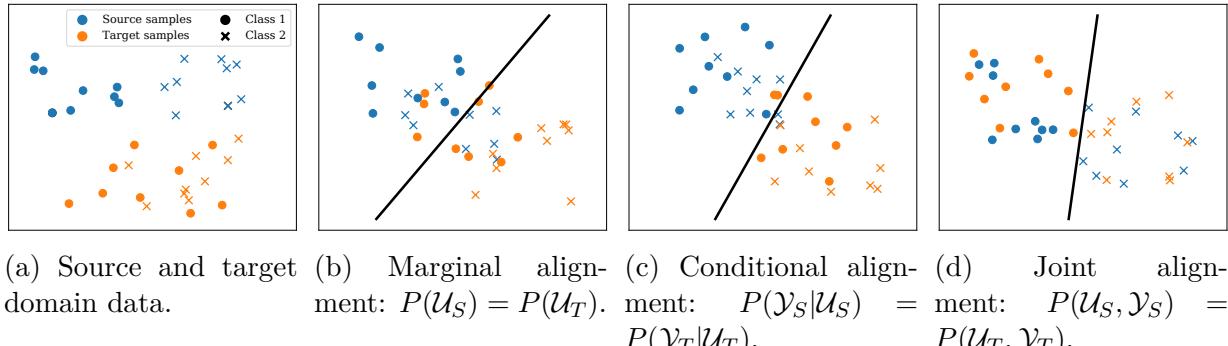


Figure 1.14: Differentiation between marginal, conditional, and joint domain alignment of source and target domain data samples (Zhang, 2021).

new deep learning-based techniques rather than conventional methodologies (e.g., feature selection, distribution adaptation, and subspace learning). Please refer to Figure 1.13 for a comprehensive overview. Within the realm of *distribution adaptation*, conventional and learning-based DA techniques can be subdivided into *marginal*, *conditional*, and *joint* adaptation (see Figure 1.14). Approaches for *marginal* distribution adaptation assume a dissimilarity between the marginal distributions of both domains, where $P(U_S) \neq P(U_T)$. These methodologies minimize the distance between the marginal distributions:

$$\min_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \approx \|P(U_S) - P(U_T)\|_2, \quad (1.12)$$

Examples of such methods are Tzeng et al. (2014); Sun et al. (2016); Ganin et al. (2016); Chen et al. (2020); Wilson et al. (2020); Liu & Xe (2021); Ott et al. (2022a). Techniques for *conditional* distribution alignment assume a difference between the conditional distributions of both domains, where $P(Y_S|U_S) \neq P(Y_T|U_T)$. These methodologies minimize the distance metric given by

$$\min_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \approx \|P(Y_S|U_S) - P(Y_T|U_T)\|_2. \quad (1.13)$$

To account for the unlabeled target domain, these methods either utilize pseudo-labels or adopt a semi-supervised setup. For *joint* distribution alignment methods, it holds $P(Y_S, U_S) \neq P(Y_T, U_T)$, where such methods minimize the joint distribution of both domains by:

$$\min_{\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) \approx \|P(U_S) - P(U_T)\|_2 + \|P(Y_S|U_S) - P(Y_T|U_T)\|_2. \quad (1.14)$$

Examples of such methods are those by Long et al. (2018); Shu et al. (2018); Rahman et al. (2020); Zhu et al. (2020b), which combine techniques from marginal and conditional distribution alignment. According to Stojanov et al. (2021), a fixed mapping of input data to the latent representation may not be appropriate for DA when using invariant features. To ensure that the latent representation does not lose valuable information, it is

essential to consider domain-specific information, which requires separate encoders for each domain. However, there is still no systemic approach to ensure that the learned marginally invariant representation contains enough semantic information and can achieve conditional invariance.

Deep learning-based methods for DA can be classified into several categories, such as discrepancy-based, adversarial-based, pseudo-labeling-based, and attention-based methods. *Adversarial*-based methods utilize an additional domain discriminator to differentiate the source domain from the target domain, as employed by several previous studies (Ganin et al., 2016; Long et al., 2018; Shu et al., 2018; Wilson et al., 2020; Liu & Xe, 2021). However, this thesis focuses on *discrepancy*-based methods that aim to minimize the distance between the source and target by incorporating distance loss functions in the activation layers of networks (Zhang, 2021), as they are the most commonly used techniques in DA applications. Commonly used distance functions include mean squared error (see Section 1.6.1) and distribution-based functions such as the Kullback-Leibler divergence or Jensen-Shannon divergence (see Section 1.6.3), as utilized in works such as Zhuang et al. (2018); Jiang et al. (2020a). Maximum mean discrepancy (MMD) is a well-known and frequently used metric (see Section 1.6.4), first proposed by Tzeng et al. (2014). Correlation alignment (CORAL) (see Section 1.6.6) by Sun et al. (2016) and higher-order moment matching (HoMM) (see Section 1.6.7) by Chen et al. (2020) are related to MMD and aim to align higher-order statistics of features. Rahman et al. (2020) combine these distances in one loss function (see Section 1.6.5). Recently, many DA methods have adopted the Wasserstein distance (Kantorovitch, 2006), such as the Earth Mover’s distance, and Sinkhorn (Courty et al., 2016) has been used to iteratively solve the Wasserstein distance and provide good geometrical properties (see Section 1.5.4). In Chapter 2, we benchmark all of these discrepancy-based methods on time-series DA classification tasks.

1.5.4 Domain Adaptation with Optimal Transport

Optimal transport (OT) is a mathematical framework that deals with the problem of finding the most efficient way to transport resources from one location to another, commonly used to model the transportation of people, goods, or information. OT has found applications in diverse fields such as fluid dynamics, image processing, and machine learning. Particularly in the context of machine learning, OT has proven to be an effective technique for measuring distances between probability distributions, and has been successfully applied in DA for computer vision tasks. However, despite this success, it has been rarely utilized for time-series classification tasks. This section aims to introduce the OT problem for DA, discuss the properties of OT and Sinkhorn, and provide a summary of relevant prior work in this area.

Optimal Transport for DA. In the following, we repeat the OT definition as described in Ott et al. (2022a) to present properties of OT and Sinkhorn. We still assume the definition of input data (see Definition 1.3.1) and feature embeddings (see Definition 1.3.3). We assume that the domain shift (present in the corresponding application) is due to an

unknown, possibly nonlinear map $\mathbf{T} : \mathcal{D}_S \rightarrow \mathcal{D}_T$ of the input space that preserves the conditional distribution $P_S(\mathcal{Y}|f(\mathbf{U}_S)) = P_T(\mathcal{Y}|\mathbf{T}(f(\mathbf{U}_S)))$ such that the label information is preserved (Courty et al., 2016). $P_S(\mathcal{Y}|\cdot)$ and $P_T(\mathcal{Y}|\cdot)$ are the conditional distributions of the source and target domains. In the case of class imbalance, it holds $P_S(\mathcal{Y}) \neq P_T(\mathcal{Y})$ and $P_S(f(\mathbf{U}_S))|\mathcal{Y}) = P_T(f(\mathbf{U}_T))|\mathcal{Y})$. For covariate shift, the conditional distributions are equal: $P_S(\mathcal{Y}|f(\mathbf{U}_S)) = P_T(\mathcal{Y}|f(\mathbf{U}_T))$. Searching for \mathbf{T} in the space of all possible transformations is intractable. Hence, \mathbf{T} is chosen such that a transportation cost

$$C(\mathbf{T}) = \int_{\mathcal{D}_S} c(f(\mathbf{U}), \mathbf{T}(f(\mathbf{U}))) d\mu(f(\mathbf{U})), \quad (1.15)$$

is minimized, where $c(a, b) : \mathcal{D}_S \times \mathcal{D}_T \rightarrow \mathbb{R}^+$ is a symmetric positive distance function over the metric space $\mathcal{D}_S \times \mathcal{D}_T$ (we assume $c(a, a) = 0$) where a, b are input data, and μ is the marginal (Courty et al., 2016; Ott et al., 2022a). \mathcal{D}_S is the source domain and \mathcal{D}_T the target domain. The optimal transportation problem is

$$\mathbf{T}_0 = \arg \min_{\mathbf{T}} \int_{\mathcal{D}_S} c(f(\mathbf{U}), \mathbf{T}(f(\mathbf{U}))) d\mu(f(\mathbf{U})). \quad (1.16)$$

This is also known as the Kantorovitch formulation (Kantorovitch, 2006). For more information about the properties of the Kantorovitch duality, see Villani (2008). This formulation allows to search a general coupling $\alpha \in \Theta$ by the *transportation plan* (Santambrogio, 2015):

$$\alpha_0 = \arg \min_{\alpha \in \Theta} \int_{\mathcal{D}_S \times \mathcal{D}_T} c(f(\mathbf{U}_S), f(\mathbf{U}_T)) d\alpha(f(\mathbf{U}_S), f(\mathbf{U}_T)), \quad (1.17)$$

where Θ is a set of all probabilistic couplings $\Theta \in P(\mathcal{D}_S \times \mathcal{D}_T)$ with marginals μ_S and μ_T .

Definition 1.5.2 (Wasserstein distance, Courty et al. (2016)). *The Wasserstein distance of order p between the marginals μ_S and μ_T can be defined as*

$$W_p(\mu_S, \mu_T) := \left(\inf_{\alpha \in \Theta} \int_{\mathcal{D}_S \times \mathcal{D}_T} d(f(\mathbf{U}_S), f(\mathbf{U}_T))^p d\alpha(f(\mathbf{U}_T), f(\mathbf{U}_S)) \right)^{\frac{1}{p}}, \quad (1.18)$$

where d is a distance metric as the cost function:

$$c(f(\mathbf{U}_T), f(\mathbf{U}_S)) = d(f(\mathbf{U}_T), f(\mathbf{U}_S))^p. \quad (1.19)$$

The Wasserstein distance is referred to as Earth Mover's distance (EMD) for the order $p = 1$ and as the quadratic Wasserstein-2 distance for order $p = 2$. Entropic regularization has recently emerged as a computationally efficient way of approximating OT costs. Entropic regularization was first introduced by Léonard (2014) and can be utilized to efficiently approximate the OT cost. We can define the regularized OT in the following.

Definition 1.5.3 (OT with entropic regularization, (Feydy et al., 2019)). *For the regularization parameter $\epsilon > 0$, the approximated OT cost is defined as*

$$\text{OT}_\epsilon(\mu_S, \mu_T) := \min_{\alpha \in \Theta} \int_{\mathcal{D}_S \times \mathcal{D}_T} c(f(\mathbf{U}_S), f(\mathbf{U}_T)) d\alpha(f(\mathbf{U}_S), f(\mathbf{U}_T)) + \epsilon \text{KL}(\alpha \|\mu_S \otimes \mu_T), \quad (1.20)$$

where μ_S and μ_T are the marginals, c is a cost function, and \otimes is the Kronecker product. The minimization is performed over the coupling measures $\Theta \in P(\mathcal{D}_S \times \mathcal{D}_T)$.

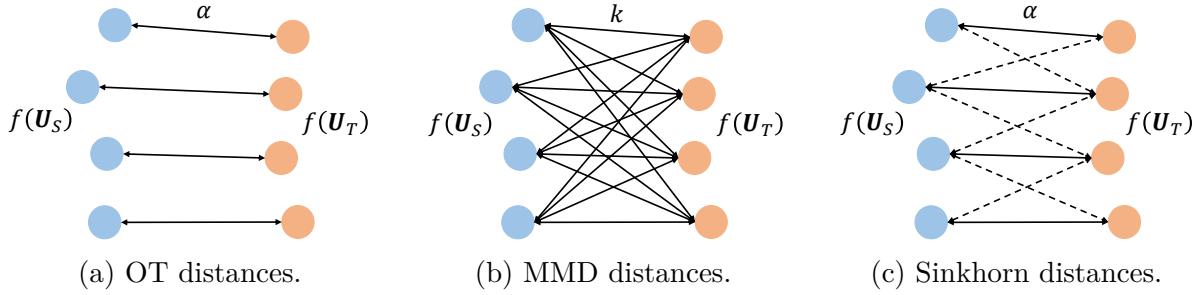


Figure 1.15: Visualization of Sinkhorn that interpolates between OT and MMD distances (Genevay et al., 2018; Feydy et al., 2019).

The cost function c in Equation 1.19 and Equation 1.20 is typically $c(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \|f(\mathbf{U}_S) - f(\mathbf{U}_T)\|^p$. KL is the Kullback-Leibler divergence defined in Section 1.6.3. Setting $\epsilon = 0$ allows to retrieve the Earth Mover’s distance for $p = 1$. The smooth problem in Equation 1.20 can be solved efficiently on the GPU for $\epsilon > 0$, see Cuturi (2013).

Definition 1.5.4 (Sinkhorn divergences, (Genevay et al., 2018)). *The Sinkhorn divergence $S_\epsilon(\mu_S, \mu_T)$ based on the cost OT_ϵ is defined as*

$$S_\epsilon(\mu_S, \mu_T) := OT_\epsilon(\mu_S, \mu_T) - \frac{1}{2} OT_\epsilon(\mu_S, \mu_S) - \frac{1}{2} OT_\epsilon(\mu_T, \mu_T). \quad (1.21)$$

It holds $S_\epsilon(\mu_S, \mu_S) = 0$ and $S_\epsilon(\mu_T, \mu_T) = 0$.

Properties of OT and Sinkhorn. From Definition 1.5.4, we can formulate two properties of the Sinkhorn divergence: (1) For $\epsilon \rightarrow 0$ it holds $S_\epsilon(\mu_S, \mu_T) \rightarrow OT_\epsilon(\mu_S, \mu_T)$. (2) As $\epsilon \rightarrow +\infty$ it holds $S_\epsilon(\mu_S, \mu_T) \rightarrow \frac{1}{2} \|\mu_S - \mu_T\|_{-c}^2$, where $\frac{1}{2} \|\mu_T - \mu_S\|_{-c}^2$ is the MMD distance (see Section 1.6.4) whose kernel is the cost from the OT problem. Hence, Sinkhorn losses have the geometric property to interpolate between MMD and OT (Genevay et al., 2018; Feydy et al., 2019), please refer to Figure 1.15. Furthermore, Sinkhorn is a convex, smooth, positive definite loss function, and hence, S_ϵ can be used as a reliable loss function, independent on the value of ϵ (Thornton & Cuturi, 2023). For the proof of Sinkhorn properties, see Feydy et al. (2019). By tuning ϵ , the regularized Sinkhorn loss is able to minimize the best of both loss functions, to utilize the non-flat geometry of OT with the high-dimensional rigidity of MMD losses (Genevay et al., 2018).

Application in Research & Contributing Papers. In their work, Courty et al. (2016) introduced Sinkhorn as a method for performing supervised DA on visual datasets, such as those used for digit, face, and object recognition. Sinkhorn was found to be effective at producing domain-invariant features, leading to improved performance. Genevay et al. (2018) evaluated the effectiveness of Sinkhorn for generative models in classification tasks, utilizing datasets such as CIFAR10 and CelebA. They found that a model incorporating

a Sinkhorn loss and large regularization outperforms both MMD and a method closer to OT (using a small ϵ) in term of classification accuracy. Thornton & Cuturi (2023) asserted that the inclusion of an entropic smoothing term in OT solvers yields differentiable, faster, more resilient-to-outliers, and more parallelizable outcomes. Several research studies aim to reduce the runtime of OT through various means such as momentum, annealing, or acceleration. Thornton & Cuturi (2023) demonstrated that an appropriately selected initialization of the Sinkhorn algorithm can lead to a significant acceleration in computation. The field of OT is vast and encompasses various directions. Therefore, to maintain the focus of the thesis, we do not discuss other related works. Instead, in our contribution presented in Ott et al. (2022a), we utilized the Earth Mover’s distance and Sinkhorn transport with and without regularization to transform left-handed embeddings into right-handed counterparts. We found that the Python Optimal Transport (POT) package⁴ (Flamary et al., 2021) was efficient and user-friendly for our purposes. Our objective was to find an optimal distance metric for the cost function presented in Equation 1.19. To this end, we evaluated 18 different distance metrics, including (squared) Euclidean, Bray Curtis, Canberra, Chebyshev, Cosine, Jaccard, Jensen-Shannon, and other distances measures. In our previous work (Ott et al., 2022c), we utilized the Earth Mover’s distance as a distance metric to compare the ground truth and predicted trajectories. In Chapter 2, we expand on this research and further evaluate the effectiveness of Sinkhorn transport for the DA task on time-series data.

1.6 Representation Learning

In the fields of machine learning and deep learning, *representation learning*, also referred to as feature learning, is a set of techniques that enable a system to automatically investigate and discover representations for a given task, thereby enhancing the performance of classification, regression, or clustering algorithms. The significance of representation learning techniques stems from the fact that the input data is typically high-dimensional and large-scale, and hence, presenting substantial mathematical and computational complexities that require resolution. Representation learning is a widely recognized technique in the fields of supervised, unsupervised, and self-supervised learning. A model has the ability to extract features from input data, and subsequently use those features to accomplish the desired task either by relying on distance constraints or by utilizing fine decomposition of instances in complete samples (Goodfellow et al., 2016). When training a model, it is imperative that the target objective, such as cross-entropy loss, and the representation learning loss are taken into consideration (Ben-David et al., 2009). In particular, the representation learning process for time-series data requires special attention, as the objective function must take into account the spatio-temporal component of the time-series data (Lafabregue et al., 2021).

An effective representation encompasses the posterior distribution of the underlying explanatory factor for the given input data. Figure 1.16 visualizes the encoding of features

⁴Python Optimal Transport package (Flamary et al., 2021): <https://pythonot.github.io/>

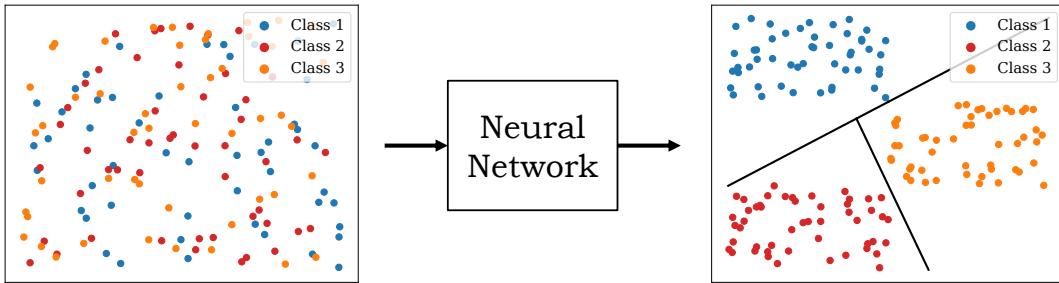


Figure 1.16: The default representation of data (left) does not encode a semantic meaning. A neural network encodes representations or features that provide a meaning related to the current task (right). Here, the task is to classify the input data (dots). In the right representation, similar data samples are closer to each other, and hence, the task is easier to solve (Goodfellow et al., 2016).

from input data by a neural network into a high-quality representation, which facilitates problem-solving properties. The primary inquiry that we aim to address in this section pertains to identifying suitable objectives for learning optimal representations. As introduced by Bengio et al. (2014), the exploration of representations is significant because they have the potential to express several non-task-specific priors, as outlined in the following: (1) Simple parametric models, such as linear models, are often incapable of capturing the complexity present in the data. As a result, many ML models rely on smoothness-based learners, such as the Gaussian kernel, in conjunction with representation learning, to achieve the desired level of performance. (2) Due to the expressive nature of representations, a vast number of possible configurations can be learned. (3) The depth of architectures is a critical factor, as features can be reused to enhance the model’s overall performance. (4) While the objective is straightforward for classification tasks (i.e., reducing the number of misclassifications), the objective is far from the ultimate objective when it comes to representation learning. (5) The model should learn representations that can disentangle the factors of variation, meaning that features are insensitive to variation in the data. (Bengio et al., 2014)

Similarity learning and deep metric learning (DML) are related fields to representation learning. The objective is to learn a similarity, metric, or distance function that quantifies the degree of similarity or relatedness between two objects, such as the features of a sub-domain of the objects’ domain. This is accomplished by learning a weight matrix that reduces the distances between similar points and increases the distances between dissimilar points. As a result, the variance of similar points decreases, while the variance of dissimilar points increases. A metric function must satisfy the following four axioms.

Definition 1.6.1 (Distance metric axioms, Ghojogh et al. (2022)). *Consider the metric space \mathcal{H} . A distance metric is a mapping $d : \mathcal{H} \times \mathcal{H} \rightarrow [0, \infty)$, which satisfies the following properties for $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k \in \mathcal{H}$:*

$$\text{non-negativity : } d(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \quad (1.22)$$

$$identity\ of\ indiscernibles : \ d(\mathbf{x}_i, \mathbf{x}_j) = 0 \Leftrightarrow \mathbf{x}_i = \mathbf{x}_j \quad (1.23)$$

$$symmetry : \ d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i) \quad (1.24)$$

$$triangle\ inequality : \ d(\mathbf{x}_i, \mathbf{x}_j) \leq d(\mathbf{x}_i, \mathbf{x}_k) + d(\mathbf{x}_k, \mathbf{x}_j) \quad (1.25)$$

Lemma 1.6.1 (Triangle inequality, Ghojogh et al. (2022)). *The triangle inequality*

$$\|\mathbf{x}_i + \mathbf{x}_j\| \leq \|\mathbf{x}_i\| + \|\mathbf{x}_j\|. \quad (1.26)$$

is satisfied, where $\|\cdot\|$ is a norm.

Proof. We can proof Lemma 1.6.1 using the squared norm in the following:

$$\begin{aligned} \|\mathbf{x}_i + \mathbf{x}_j\|^2 &= (\mathbf{x}_i + \mathbf{x}_j)^T (\mathbf{x}_i + \mathbf{x}_j) \\ &= \|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 + 2\mathbf{x}_i^T \mathbf{x}_j \\ &\stackrel{\text{CS}}{\leq} \|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 + 2\|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\| \\ &= (\|\mathbf{x}_i\| + \|\mathbf{x}_j\|)^2, \end{aligned} \quad (1.27)$$

where $\stackrel{\text{CS}}{\leq}$ denotes the use of the Cauchy-Schwarz inequality⁵. \square

Several DML functions do not adhere to the axiom of identity of indiscernibles and instead learn a pseudo-metric. Typically, any distance metric $d(\cdot, \cdot) : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$ can be employed (Ghojogh et al., 2022). This section provides an overview of the metrics used in representation learning and their associated characteristics.

Notations & Goals. The following notations for DML and DA are valid for the whole section unless stated differently and are relevant for the contributing papers (Ott et al., 2022a,b, 2023c). We briefly repeat the definition of MTS from Section 1.2.1 and the definition of DA from Section 1.5.1. We define an MTS \mathbf{U} (see Definition 1.3.1). The MTS training set is a subset of the array \mathcal{U} . The aim of MTS classification is to predict an unknown class label $y \in \mathcal{Y}$ for a given MTS. For the setup of DA, we further define domain-specific notations. A domain \mathcal{D} consists of a feature space \mathcal{U} with marginal probability $P(\mathcal{U})$. The task is defined by the label space \mathcal{Y} . We define the source domain dataset as \mathcal{U}_S with MTS \mathbf{U}_S with $n_S \in \mathbb{N}$ samples, and the target domain dataset as \mathcal{U}_T with MTS \mathbf{U}_T and $n_T \in \mathbb{N}$ samples. We denote one specific source domain sample as \mathbf{U}_S^i with $i \in \{1, \dots, n_S\}$, and one specific target domain sample as \mathbf{U}_T^i with $i \in \{1, \dots, n_T\}$. When considering MTS classification, there is a source domain \mathcal{D}_S of $n_S \in \mathbb{N}$ labeled samples of $|\mathcal{Y}_S^i| \in \mathbb{N}$ categories, and a target domain \mathcal{D}_T of $n_T \in \mathbb{N}$ labeled samples of $|\mathcal{Y}_T^i| \in \mathbb{N}$ categories (see Section 1.5.1). In the field of CMR, the target domain dataset can be data from one modality, while the source domain dataset is from another modality, with a domain shift between source and target domain datasets. The source domain embedding of the neural network is defined by $f(\mathbf{U}_S) \in \mathbb{R}^{q_S \times w_S}$ for the corresponding MTS sample

⁵Cauchy-Schwarz inequality: $\mathbf{x}_i^T \mathbf{x}_j \leq \|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\|$.

\mathbf{U}_S , respectively the target domain embedding $f(\mathbf{U}_T) \in \mathbb{R}^{q_T \times w_T}$ for the corresponding target domain sample \mathbf{U}_T (see Definition 1.3.3). $q_S \times w_S$ and $q_T \times w_T$ are the sizes of the output of the neural network layer, and it holds: $q_S = q_T$ and $w_S = w_T$. From this, we can formulate two goals: (1) Given the smaller adaptation set of the source domain \mathcal{U}_S with MTS \mathbf{U}_S , the goal of DA is to find an optimal transformation \mathbf{T} of the representation of the latent embedding $f(\mathbf{U}_S)$ of the source domain to the representation of the latent embedding $f(\mathbf{U}_T)$ of the target domain, such that the prediction of the unknown class label y_S of the source domain is maximized. (2) Given the metric space \mathcal{H} , we want to compute the distance by the DML distance metric $d : \mathcal{H} \times \mathcal{H} \rightarrow [0, \infty)$ (optionally, with respect to the axioms in Definition 1.6.1) between the source domain embeddings $f(\mathbf{U}_S)$ and the target domain embeddings $f(\mathbf{U}_T)$. The goal is to minimize the distance and align the statistics of both embeddings, and hence, reduce the domain shift in the DA setting, or find an optimal embedding between two or more modalities in the setting of CMR. For definitions of distance metrics between general vectors, we use the notation of the vectors $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^n$ with size n . We denote the covariance matrix as $\mathbf{C} = \text{Cov}(\mathbf{x}, \mathbf{y})$ between the vectors \mathbf{x} and \mathbf{y} , and the standard deviation as σ_x of the vector \mathbf{x} , respectively for \mathbf{y} . $\bar{\mathbf{x}}$ is the mean of \mathbf{x} , respectively for \mathbf{y} .

Outlook. In this section, we begin by providing an overview of the mean squared error (Section 1.6.1) and the Cosine similarity and Pearson correlation (Section 1.6.2), with the Cosine similarity commonly employed for cross-modal learning. For completeness, we also present the Kullback-Leibler divergence (Section 1.6.3). Following this, we introduce the maximum mean discrepancy and maximum covariance discrepancy in Section 1.6.4 and Section 1.6.5, respectively. Section 1.6.6 provides an overview of second-order discrepancy metrics, specifically correlation alignment, while Section 1.6.7 presents higher order metrics.

1.6.1 Mean Squared Error & Frobenius Norm

The mean squared error (MSE), also known as the Euclidean distance, is the measure between two one-dimensional vectors $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^n$ of length n , and is defined as

$$\text{MSE}(\mathbf{x}, \mathbf{y}) := \frac{1}{n} \|\mathbf{x} - \mathbf{y}\|_2^2 = \frac{1}{n} \sum_{i=1}^n n(\mathbf{x}_i - \mathbf{y}_i)^2. \quad (1.28)$$

The root mean squared error (RMSE) is the square root of the MSE defined as

$$\text{RMSE}(\mathbf{x}, \mathbf{y}) := \sqrt{\frac{1}{n} \|\mathbf{x} - \mathbf{y}\|_2^2} = \sqrt{\frac{1}{n} \sum_{i=1}^n n(\mathbf{x}_i - \mathbf{y}_i)^2}. \quad (1.29)$$

The axioms 1.22 to 1.25 are fulfilled by both the MSE and RMSE. It is customary to evaluate the distance between feature embeddings or statistical features of these embeddings represented as two-dimensional matrices, as defined previously, by computing the

Frobenius norm between the matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{m \times n}$,

$$F(\mathbf{A}, \mathbf{B}) := \sqrt{\|\mathbf{A} - \mathbf{B}\|_F^2} = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |A_{ij} - B_{ij}|^2}, \quad (1.30)$$

see Trefethen & Bau (1997). Numerous alternatives to the MSE exist that are less sensitive to outliers, although they may have steep gradients near the optimal point and rely on hyperparameters. Examples of such alternatives include the mean absolute error (MAE), the Huber metric, the Andrew's Sine metric, the Bray Curtis metric, the Tukey's Biweight metric, the Poisson metric, and others as noted by Huber & Ronchetti (2009).

Application in Research. The MSE is a commonly utilized measure in various CMR or contrastive learning tasks, such as face recognition (Schroff et al., 2015), speech classification (Bredin, 2017), zero-shot learning of sketches (Chaudhuri et al., 2020), multimedia retrieval (Huang et al., 2020), visual semantic embedding (Chun et al., 2021), and signature verification (Wan & Zou, 2021), among others.

Application in Contributing Papers. In Ott et al. (2022c), we utilized the MSE metric as a means of computing the distance between predicted trajectories and their corresponding ground truth trajectory, using a differentiable loss function of MSE. We also compared its performance to that of the Andrew's Sine and Huber metrics. Conversely, in Ott et al. (2023c), we used the MSE metric as a baseline for computing the distance between image and time-series handwritten data in order to align features within a neural network. This application yielded lower performance results for the MSE metric, prompting us to exclude it from the contributing papers (Ott et al., 2022a,b). In addition, we compared the MSE to the Bray Curtis and Poisson variants on a synthetically generated dataset. It is noteworthy that in recent years, the MSE is becoming less prevalent for CMR tasks, with the Cosine similarity being a more commonly utilized metric, as described in the subsequent section.

1.6.2 Cosine Similarity & Pearson Correlation

The cosine of two non-zero vectors $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^n$ of size n can be obtained using the Euclidean dot product formula, which is given by $\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \cdot \|\mathbf{y}\| \cdot \cos(\theta)$. The Cosine similarity (CS), $\cos(\theta)$, is then represented as follows:

$$CS(\mathbf{x}, \mathbf{y}) := \cos(\theta) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|}, \quad (1.31)$$

where the dot product of two vectors is defined as $\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^n x_i y_i$, and θ denotes the angle between \mathbf{x} and \mathbf{y} . The range of the CS is $[-1, 1]$, where $CS = -1$ implies that the two vectors are exact opposites, $CS = 1$ implies that the two vectors are exactly the same,

and $\text{CS} = 0$ indicates orthogonality⁶ or decorrelation. As CS is a measure of similarity, the cosine distance is the complement of the CS, with

$$\text{cosine distance}(\mathbf{x}, \mathbf{y}) := 1 - \text{CS}(\mathbf{x}, \mathbf{y}), \quad (1.32)$$

in the positive space $[0, 1]$. The objective of neural networks is to maximize the CS, which is equivalent to minimizing the cosine distance as loss function. However, the cosine distance does not satisfy the triangular inequality $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (axiom 1.25) or the Cauchy-Schwarz inequality $|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\|$. Hence, the cosine distance is not considered a valid distance metric (Giller, 2012).

The Pearson correlation (PC), introduced by Pearson (1920), is invariant to shifts, and is defined by

$$\text{PC}(\mathbf{x}, \mathbf{y}) := \frac{\text{Cov}(\mathbf{x}, \mathbf{y})}{\sigma_{\mathbf{x}} \sigma_{\mathbf{y}}} = \frac{(\mathbf{x} - \bar{\mathbf{x}}) \cdot (\mathbf{y} - \bar{\mathbf{y}})}{\|(\mathbf{x} - \bar{\mathbf{x}})\| \cdot \|(\mathbf{y} - \bar{\mathbf{y}})\|}, \quad (1.33)$$

where $\text{Cov}(\mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{y}_i - \bar{\mathbf{y}})$ is the covariance between \mathbf{x} and \mathbf{y} , $\sigma_{\mathbf{x}}$ is the standard deviation of \mathbf{x} , respectively for \mathbf{y} , with $\sigma_{\mathbf{x}}^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})^2$, and the mean $\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$ of the vector \mathbf{x} , respectively for \mathbf{y} . Then, we minimize the Pearson distance

$$\text{Pearson distance}(\mathbf{x}, \mathbf{y}) := 1 - \frac{\text{Cov}(\mathbf{x}, \mathbf{y})}{\sigma_{\mathbf{x}} \sigma_{\mathbf{y}}}. \quad (1.34)$$

The Pearson correlation is a measure of linear correlation and is in the range $[-1, 1]$.

For completeness, the concordance correlation coefficient (CCC), proposed by Lin (1989), is a measure agreement between two vectors, and is defined by

$$\text{CCC}(\mathbf{x}, \mathbf{y}) := \frac{2\text{PC}(\mathbf{x}, \mathbf{y})\sigma_{\mathbf{x}}\sigma_{\mathbf{y}}}{\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{y}}^2 + (\bar{\mathbf{x}} - \bar{\mathbf{y}})^2}. \quad (1.35)$$

The use of biased or unbiased estimates is relevant for the concordance correlation when compared to the Pearson correlation.

Application in Research. The CS metric finds its application in various multimedia retrieval tasks, including text matching techniques (Wu et al., 2008) where it normalizes the document length during comparison. It is also used in visual semantic embedding (Wang et al., 2022; Singh et al., 2022), human activity recognition (Jain et al., 2022), speech and audio extraction (Ohishi et al., 2022; Guzhov et al., 2022), and time-series classification (Deldari et al., 2022b). In contrast, the PC is seldom used in CMR, except for the work by Zhu et al. (2020a) where the cross-modal correlation between texts and images is evaluated using the PC. The CCC has not been widely explored recently, and apart from Carrasco & Jover (2003), there are no recent publications on this metric.

⁶Orthogonality means the perpendicularity between two vectors \mathbf{x} and \mathbf{y} , that is given if their inner product $\langle \mathbf{x}, \mathbf{y} \rangle$ is zero.

Application in Contributing Papers. In the contributing paper (Ott et al., 2022c), we assessed the performance of both the CS and PC coefficients for trajectory alignment. Furthermore, in the contributing papers (Ott et al., 2022a,b, 2023c) and in Chapter 2, we utilized these correlation coefficients to align features between neural networks.

1.6.3 Kullback-Leibler & Jensen-Shannon Divergence

The Kullback-Leibler (KL) divergence (Kullback & Leibler, 1951) is a mathematical measure of dissimilarity between two probability distributions, denoted as P and Q . Specifically, for the discrete probability distributions P and Q over the probability space \mathcal{H} , the relative entropy is given by

$$\text{KL}(P||Q) := \sum_{h \in \mathcal{H}} P(h) \log \left(\frac{P(h)}{Q(h)} \right), \quad (1.36)$$

that is not symmetric and does not satisfy the triangle inequality. In order to address this issue, the Jensen-Shannon divergence (JSD) is defined by

$$\text{JSD}(P||Q) := \frac{1}{2} \text{KL}(P||M) + \frac{1}{2} \text{KL}(Q||M), \quad (1.37)$$

with $M = \frac{1}{2}(P + Q)$ being the average distribution. JSD is a symmetric and smoothed version of the KL divergence. While simple norms and divergences such as relative entropy and total variation can only compare densities point-wise, they are unable to capture the geometric properties of the problem and can be unstable when the distributions' supports are deformed. On the other hand, optimal transport (discussed in Section 1.5.4) and maximum mean discrepancy (described in the following Section 1.6.4) take into account the underlying space's geometry and are thus better suited for capturing these properties (Feydy et al., 2019).

Application in Research & Contributing Papers. The KL divergence is a widely used measure in variational autoencoder (VAE) applications, as shown in Lin et al. (2020). However, it is less frequently used in CMR applications, as noted in Chen et al. (2021c). In our study (Ott et al., 2023c), we conducted an evaluation of the KL divergence in this context and we conducted experiments utilizing KL and JSD for DA in Chapter 2.

1.6.4 Maximum Mean Discrepancy

The concept of maximum mean discrepancy (MMD) was initially introduced by Borgwardt et al. (2006). Consider data sampled from two unknown distributions with densities p and q (Borel probability distributions), and defined in $\mathcal{U} \subset \mathbb{R}^d$. Let $\mathcal{U}_S = \{f(\mathbf{U}_S^i) \sim p, \text{i.i.d.}, i = 1, \dots, n_S\}$ be the dataset from the source domain and $\mathcal{U}_T = \{f(\mathbf{U}_T^i) \sim q, \text{i.i.d.}, i = 1, \dots, n_T\}$ be the dataset of the target domain. The samples are composed of independent

and identically distributed (i.i.d.) observations. The goal is to test whether the two datasets follow the same distribution, which is equivalent to performing the hypothesis test $H_0 : p = q$ versus $H_1 : p \neq q$. Let \mathcal{G} be a class of functions $g : \mathcal{U} \rightarrow \mathbb{R}$. The MMD is defined by Gretton et al. (2012) as follows:

$$\text{MMD}(\mathcal{G}, p, q) = \sup_{g \in \mathcal{G}} \left(\mathbb{E}_{\mathcal{U}_S \sim p} [g(f(\mathbf{U}_S))] - \mathbb{E}_{\mathcal{U}_T \sim q} [g(f(\mathbf{U}_T))] \right). \quad (1.38)$$

This is known as an integral probability metric. A biased empirical estimate⁷ of the MMD is then defined as

$$\text{MMD}_b(\mathcal{G}, p, q) = \sup_{g \in \mathcal{G}} \left(\frac{1}{n_S} \sum_{i=1}^{n_S} g(f(\mathbf{U}_S^i)) - \frac{1}{n_T} \sum_{i=1}^{n_T} g(f(\mathbf{U}_T^i)) \right). \quad (1.39)$$

Linear MMD as Loss Function. Tzeng et al. (2014) applied the MMD for domain adaptation in CNNs. To compute the MMD loss between the source domain features $f(\mathbf{U}_S)$ and the target domain features $f(\mathbf{U}_T)$, they used the equation:

$$\mathcal{L}_{\text{MMD}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \left\| \frac{1}{n_S} \sum_{i=1}^{n_S} f(\mathbf{U}_S^i) - \frac{1}{n_T} \sum_{i=1}^{n_T} f(\mathbf{U}_T^i) \right\|^2, \quad (1.40)$$

where n_S and n_T are the number of source and target samples in the mini-batch, respectively. MMD norms are scalable for large batches, computationally efficient, and require lower sample complexity than optimal transport (Feydy et al., 2019). Tzeng et al. (2014) conducted experiments to evaluate a model for DA in CNNs based on two selection choices of an adaptation layer and the domain distance loss: (1) The layer in the network to place the adaptation layer. (2) The dimension of the layer to choose. The results indicate an inverse relationship between the MMD computed between the source and target samples and the accuracy on the target domain test dataset. They achieved the highest classification accuracy with the second-to-last adaptation layer. Furthermore, they observed an inverse relationship between MMD and the accuracy on the test dataset for different values of adaptation layer dimensionality.

MMD in RKHS. According to Borgwardt et al. (2006); Long et al. (2017), it holds $p = q$ iff $\text{MMD}(p, q) = 0$ (if \mathcal{G} is rich enough). If the class of functions \mathcal{G} is too rich, the resulting $\text{MMD}(p, q)$ may produce significantly different results from zero. To avoid overfitting, the function class can be restricted. One approach to restricting \mathcal{G} is to choose it to be the unit ball in a universal kernel Hilbert space (RKHS).

Definition 1.6.2 (MMD in RKHS, Borgwardt et al. (2006)). *We let \mathcal{H} be a complete inner product space (i.e., a Hilbert space) with unit ball \mathcal{G} and let p, q be Borel probability measures. The MMD is defined as the distance between the mean embeddings of the two distributions by*

$$\text{MMD}^2(\mathcal{G}, p, q) = \|\mu_{\mathcal{U}_S}(p) - \mu_{\mathcal{U}_T}(q)\|_{\mathcal{H}}^2. \quad (1.41)$$

⁷The population expectations are replaced with empirical expectations.

Proof. We can proof the definition of $\text{MMD}^2(\mathcal{G}, p, q)$ according to Gretton et al. (2012) in the following:

$$\begin{aligned}\text{MMD}^2(\mathcal{G}, p, q) &= \left[\sup_{\|g\|_{\mathcal{H}} \leq 1} \left(\mathbb{E}_{\mathcal{U}_S \sim p} [g(f(\mathbf{U}_S))] - \mathbb{E}_{\mathcal{U}_T \sim q} [g(f(\mathbf{U}_T))] \right) \right]^2 \\ &= \left[\sup_{\|g\|_{\mathcal{H}} \leq 1} \left\langle \mu_{\mathcal{U}_S}(p) - \mu_{\mathcal{U}_T}(q), g \right\rangle_{\mathcal{H}} \right]^2 \\ &= \|\mu_{\mathcal{U}_S}(p) - \mu_{\mathcal{U}_T}(q)\|_{\mathcal{H}}^2.\end{aligned}\tag{1.42}$$

□

The kernelized MMD (kMMD) considers test functions in the RKHS of positive semi-definite kernel $k(f(\mathbf{U}_S), f(\mathbf{U}_T))$. Often, the Gaussian radial basis function (RBF) kernel $k(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \exp(-\gamma \|f(\mathbf{U}_S) - f(\mathbf{U}_T)\|_2)$ with $\gamma = \frac{1}{2\beta^2}$ is used. β is a free parameter. Then, the squared and biased empirical kMMD is defined by

$$\text{MMD}_k^2 = \int_{\mathcal{U}} \int_{\mathcal{U}} k(f(\mathbf{U}_S), f(\mathbf{U}_T)) (\bar{p} - \bar{q}) f(\mathbf{U}_T) df(\mathbf{U}_S) df(\mathbf{U}_T),\tag{1.43}$$

where $\bar{p} := \frac{1}{n_S} \sum_{i=1}^{n_S} \delta_{u_S^i}$ and $\bar{q} := \frac{1}{n_T} \sum_{i=1}^{n_T} \delta_{u_T^i}$ (Cheng & Xie, 2021). In practice, an estimate of the unbiased kMMD compares the square distance between the empirical kernel mean embeddings as

$$\begin{aligned}\text{kMMD}_u(\mathcal{G}, p, q) &= \frac{1}{n_S^2} \sum_{i=1}^{n_S} \sum_{j=1}^{n_S} k(f(\mathbf{U}_S^i), f(\mathbf{U}_S^j)) \\ &\quad + \frac{1}{n_T^2} \sum_{i=1}^{n_T} \sum_{j=1}^{n_T} k(f(\mathbf{U}_T^i), f(\mathbf{U}_T^j)) - \frac{2}{n_S n_T} \sum_{i=1}^{n_S} \sum_{j=1}^{n_T} k(f(\mathbf{U}_S^i), f(\mathbf{U}_T^j)),\end{aligned}\tag{1.44}$$

as an estimation of $\text{MMD}_k^2(\mathcal{G}, p, q)$ (Long et al., 2017).

Application in Research & Contributing Papers. Recently, Deka & Sutherland (2023) introduced MMD-B-Fair, a technique for acquiring unbiased representations of data using kernelized MMD. The approach is invariant to a binary-sensitive attribute and exhibits robust performance in adversarial learning and generative modelling. In Ott et al. (2023c), we utilized the MMD from Equation 1.39 and kMMD from Equation 1.44 to minimize the distance between features of time-series and visual embeddings from two applications: (1) Generated sinusoidal time-series data and their corresponding visual Gramian angular summation field, and (2) multivariate time-series from OnHW sensor-enhanced pens and generated images for offline handwriting recognition. Additionally, we compared MMD and kMMD in Ott et al. (2022a) for DA and in Ott et al. (2022b) for aligning embeddings between paper and tablet OnHW. Chapter 2 presents an evaluation of MMD and kMMD on time-series datasets.

1.6.5 Maximum Covariance Discrepancy & Maximum Mean and Covariance Discrepancy

Given the assumptions and notations introduced in Section 1.6.4, the maximum covariance discrepancy (MCD) can be formally defined as follows:

$$\text{MCD}_{\mathcal{H}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \|\text{Cov}(f(\mathbf{U}_S)) - \text{Cov}(f(\mathbf{U}_T))\|_{\mathcal{H}}^2, \quad (1.45)$$

where $\text{Cov}(\cdot)$ is the covariance matrix of the embeddings $f(\mathbf{U}_S)$ and $f(\mathbf{U}_T)$, respectively. We can express the loss function for the marginal distribution discrepancy based on the MCD as

$$\mathcal{L}_{\text{MCD}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \left\| \frac{1}{n_S} \sum_{i=1}^{n_S} f(\mathbf{U}_S^i) \left(\mathbb{I}_{n_S} - \frac{1}{n_S} \mathbf{1}_{n_S} \right) f(\mathbf{U}_S^i)^T - \frac{1}{n_T} \sum_{i=1}^{n_T} f(\mathbf{U}_T^i) \left(\mathbb{I}_{n_T} - \frac{1}{n_T} \mathbf{1}_{n_T} \right) f(\mathbf{U}_T^i)^T \right\|_F^2, \quad (1.46)$$

where $\|\cdot\|_F$ denotes the Frobenius norm, \mathbb{I}_{n_S} and \mathbb{I}_{n_T} are the identity matrices of size n_S or n_T , and $\mathbf{1}_{n_S}$ and $\mathbf{1}_{n_T}$ are the vectors of ones of length n_S or n_T . For more information on correlation alignment with covariance discrepancy, please refer to the following Section 1.6.6. The MMD and MCD can be incorporated into a single metric, named as maximum mean and covariance discrepancy (MMCD), using the following definition

$$\text{MMCD}_{\mathcal{H}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \beta_1 \text{MMD}_{\mathcal{H}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) + \beta_2 \text{MCD}_{\mathcal{H}}(f(\mathbf{U}_S), f(\mathbf{U}_T)), \quad (1.47)$$

based on Equation 1.40 (for linear MMD), Equation 1.44 (for kernelized MMD), and Equation 1.45 (for MCD), where β_1 and β_2 are hyperparameters to balance the impact of first-order and second-order statistical discrepancies (Zhang et al., 2020a; Alipour & Tahmoresnezhad, 2021). The MMCD is not commonly used in practical applications, and therefore, we did not use this metric in our own applications. However, we present these metrics for the sake of completeness and provide an evaluation of the linear MMCD and kernelized MMCD in Chapter 2. We will primarily focus on correlation alignment.

1.6.6 Correlation Alignment

The subspace-based method known as correlation alignment (CORAL), proposed by Sun & Saenko (2015); Sun et al. (2016), is a straightforward and computationally efficient technique. CORAL mitigates domain shift by aligning the second-order statistics (i.e., the original feature distributions of the source and target domains), without requiring any target domain labels. Unlike other subspace-based methods, CORAL aligns the original feature distributions of the embeddings instead of the bases of lower-dimensional subspace. Let μ_S and μ_T be the means of the feature embeddings $f(\mathbf{U}_S)$ and $f(\mathbf{U}_T)$ of the source and target domains, respectively, and \mathbf{C}_S and \mathbf{C}_T be the corresponding covariance matrices. The features are normalized to have zero mean ($\mu_S = \mu_T = 0$), while $\mathbf{C}_S \neq \mathbf{C}_T$. CORAL

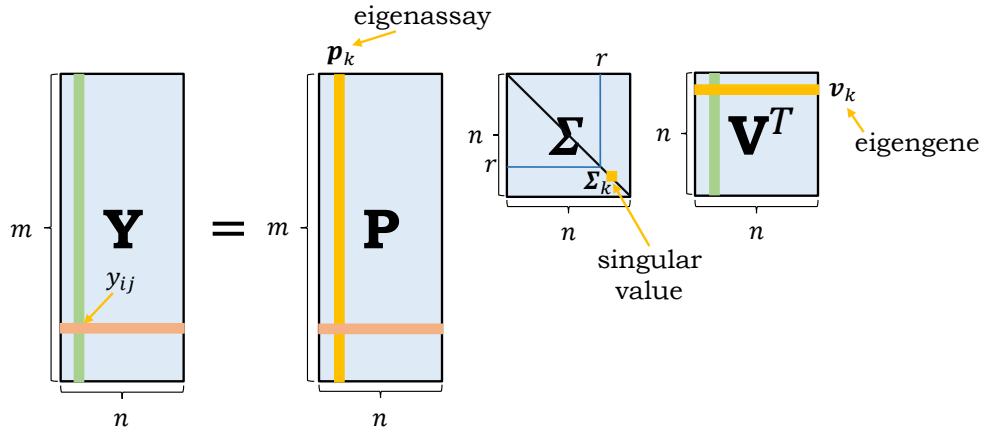


Figure 1.17: Structure of the singular value decomposition (SVD) $\mathbf{Y} = \mathbf{P}\Sigma\mathbf{V}^T$ of the matrix \mathbf{Y} with matrix lengths m and n , the singular values Σ , and the left and right singular vectors \mathbf{P} and \mathbf{V} (Trefethen & Bau, 1997).

aims to reduce the difference between $f(\mathbf{U}_S)$ and $f(\mathbf{U}_T)$ through a linear transformation \mathbf{A} by

$$\min_{\mathbf{A}} \|\mathbf{C}_{\hat{S}} - \mathbf{C}_T\|_F^2 = \min_{\mathbf{A}} \|\mathbf{A}^T \mathbf{C}_S \mathbf{A} - \mathbf{C}_T\|_F^2, \quad (1.48)$$

where $\|\cdot\|_F^2$ is the squared Frobenius norm, and $\mathbf{C}_{\hat{S}}$ is the covariance of the transformed source features $f(\mathbf{U}_S)\mathbf{A}$ (Sun et al., 2016).

Theorem 1.6.2 (Solution to the CORAL problem, Sun et al. (2016); Ott et al. (2022a)). *The optimal solution of the problem in Equation 1.48 is given by*

$$\mathbf{A}^* = \mathbf{P}_S \Sigma_S^{+ \frac{1}{2}} \mathbf{P}_S^T \mathbf{P}_{T[1:r]} \Sigma_{T[1:r]}^{\frac{1}{2}} \mathbf{P}_{T[1:r]}^T, \quad (1.49)$$

with $r = \min(r_{\mathbf{C}_S}, r_{\mathbf{C}_T})$, where $r_{\mathbf{C}_S}$ and $r_{\mathbf{C}_T}$ denote the rank of \mathbf{C}_S and \mathbf{C}_T , and Σ^+ is the Moore-Penrose pseudoinverse of Σ . With the singular value decomposition (SVD) of a real matrix \mathbf{Y} the largest $r \leq r_{\mathbf{Y}}$ singular values $\Sigma_{\mathbf{Y}_{[1:r]}}$, and left and right singular vectors $\mathbf{P}_{\mathbf{Y}_{[1:r]}}$ and $\mathbf{V}_{\mathbf{Y}_{[1:r]}}$ of $\mathbf{Y} = \mathbf{P}_{\mathbf{Y}} \Sigma_{\mathbf{Y}} \mathbf{V}_{\mathbf{Y}}^T$ of the real matrix \mathbf{Y} of rank $r_{\mathbf{Y}}$ can be computed.

Figure 1.17 demonstrates an SVD of the matrix \mathbf{Y} . For more information on the SVD, see Trefethen & Bau (1997); Cai et al. (2010).

Proof. We provide the proof according to Sun et al. (2016). $\mathbf{A}^T \mathbf{C}_S \mathbf{A}$ does not increase the rank of \mathbf{C}_S , as \mathbf{A} is a linear transformation, and hence, it holds: $r_{\mathbf{C}_{\hat{S}}} \leq r_{\mathbf{C}_S}$. The SVD on the symmetric matrices \mathbf{C}_S and \mathbf{C}_T gives $\mathbf{C}_S = \mathbf{P}_S \Sigma_S \mathbf{P}_S^T$ and $\mathbf{C}_T = \mathbf{P}_T \Sigma_T \mathbf{P}_T^T$. Two cases with respect to $r_{\mathbf{C}_S}$ and $r_{\mathbf{C}_T}$ have to be considered: (1) If $r_{\mathbf{C}_S} > r_{\mathbf{C}_T}$, the optimal solution of Equation 1.48 is $\mathbf{C}_{\hat{S}} = \mathbf{C}_T$, and hence, $\mathbf{C}_{\hat{S}} = \mathbf{P}_T \Sigma_T \mathbf{P}_T^T = \mathbf{P}_{T[1:r]} \Sigma_{T[1:r]} \mathbf{P}_{T[1:r]}^T$ with $r = r_{\mathbf{C}_T}$. (2) If $r_{\mathbf{C}_S} \leq r_{\mathbf{C}_T}$, the optimal solution to Equation 1.48 is $\mathbf{C}_{\hat{S}} = \mathbf{P}_T \Sigma_T \mathbf{P}_T^T = \mathbf{P}_{T[1:r]} \Sigma_{T[1:r]} \mathbf{P}_{T[1:r]}^T$ with $r = r_{\mathbf{C}_S}$.

When case (1) and case (2) are combined, the optimal solution is $\mathbf{C}_{\hat{S}} = \mathbf{P}_T \boldsymbol{\Sigma}_T \mathbf{P}_T^T = \mathbf{P}_{T[1:r]} \boldsymbol{\Sigma}_{T[1:r]} \mathbf{P}_{T[1:r]}^T$ with $r = \min(r_{\mathbf{C}_S}, r_{\mathbf{C}_T})$. As it holds $\mathbf{C}_{\hat{S}} = \mathbf{A}^T \mathbf{C}_S \mathbf{A}$ in Equation 1.48, it also holds

$$\mathbf{A}^T \mathbf{C}_S \mathbf{A} = \mathbf{P}_{T[1:r]} \boldsymbol{\Sigma}_{T[1:r]} \mathbf{P}_{T[1:r]}^T. \quad (1.50)$$

We can replace $\mathbf{C}_{\hat{S}}$ in Equation 1.50 with $\mathbf{P}_S \boldsymbol{\Sigma}_S \mathbf{P}_S^T$ and get

$$\mathbf{A}^T \mathbf{P}_S \boldsymbol{\Sigma}_S \mathbf{P}_S^T \mathbf{A} = \mathbf{P}_{T[1:r]} \boldsymbol{\Sigma}_{T[1:r]} \mathbf{P}_{T[1:r]}^T. \quad (1.51)$$

Rewriting Equation 1.51 gives

$$(\mathbf{P}_S^T \mathbf{A})^T \boldsymbol{\Sigma}_S (\mathbf{P}_S^T \mathbf{A}) = \mathbf{P}_{T[1:r]} \boldsymbol{\Sigma}_{T[1:r]} \mathbf{P}_{T[1:r]}^T \quad (1.52)$$

that results in

$$(\mathbf{P}_S^T \mathbf{A})^T \boldsymbol{\Sigma}_S (\mathbf{P}_S^T \mathbf{A}) = \mathbf{E}^T \boldsymbol{\Sigma}_S \mathbf{E}, \quad (1.53)$$

by defining $\mathbf{E} = \boldsymbol{\Sigma}_S^{+\frac{1}{2}} \mathbf{P}_S^T \mathbf{P}_{T[1:r]} \boldsymbol{\Sigma}_{T[1:r]}^{\frac{1}{2}} \mathbf{P}_{T[1:r]}^T$. Next, we set $\mathbf{P}_S^T \mathbf{A} = \mathbf{E}$. Then, we obtain the optimal solution of Theorem 1.6.2 as

$$\mathbf{A}^* = \mathbf{P}_S \mathbf{E} = (\mathbf{P}_S \boldsymbol{\Sigma}_S^{+\frac{1}{2}} \mathbf{P}_S^T) (\mathbf{P}_{T[1:r]} \boldsymbol{\Sigma}_{T[1:r]}^{\frac{1}{2}} \mathbf{P}_{T[1:r]}^T). \quad (1.54)$$

□

The normalization of input features play a crucial role in several ML techniques. Batch normalization, as proposed by Ioffe & Szegedy (2015), normalizes mini-batches to compensate for internal covariate shift. However, it does not account for external covariate shift. On the other hand, MMD applies the same transformation to both source and target domains, while CORAL utilizes asymmetric transformations, making CORAL more flexible as noted by Sun et al. (2016). Although CORAL is an effective and simple approach, it has two primary limitations: Firstly, it relies on a linear transformation, which may not capture the most informative feature transformations. Secondly, it performs feature extraction, covariance calculations, feature alignment, and classifier training sequentially, rather than end-to-end, as highlighted by Morerio & Murino (2017). To address these issues, the following section will introduce Deep CORAL.

Deep CORAL (Sun et al., 2016; Sun & Saenko, 2016) aims to align the second-order statistics of feature embeddings generated by a deep neural network. This is achieved through designing a differentiable loss function, referred to as the Deep CORAL loss, which is defined as the distance between the second-order statistics of the feature embeddings as follows:

$$\mathcal{L}_{\text{CORAL}} = \frac{1}{4d^2} \|\mathbf{C}_S - \mathbf{C}_T\|_F^2, \quad (1.55)$$

where $\|\cdot\|_F$ is the Frobenius norm, d is the dimension of the source and target domain features, and the covariance matrices of the source and target domains are defined by

$$\mathbf{C}_S = \frac{1}{n_S - 1} \left(f(\mathbf{U}_S)^T f(\mathbf{U}_S) - \frac{1}{n_S} (\mathbf{1}_{n_S}^T f(\mathbf{U}_S))^T (\mathbf{1}_{n_S} f(\mathbf{U}_S)) \right) \quad (1.56)$$

$$\mathbf{C}_T = \frac{1}{n_T - 1} \left(f(\mathbf{U}_T)^T f(\mathbf{U}_T) - \frac{1}{n_T} (\mathbf{1}_{n_T}^T f(\mathbf{U}_T))^T (\mathbf{1}_{n_T} f(\mathbf{U}_T)) \right), \quad (1.57)$$

where $\mathbf{1}_{n_S}$ represents a column vector of size n_S with all elements equal to 1, and similarly for $\mathbf{1}_{n_T}$, where n_S and n_T denote the number of source and target data, respectively. Since CORAL is differentiable, its gradient can be computed using the chain rule and is obtained with

$$\frac{\partial \mathcal{L}_{\text{CORAL}}}{\partial f(\mathbf{U}_S^i)^j} = \frac{1}{d^2(n_S - 1)} \left((f(\mathbf{U}_S^i)^T - \frac{1}{n_S} (\mathbf{1}_{n_S}^T f(\mathbf{U}_S^i))^T \mathbf{1}_{n_S}^T)^T (\mathbf{C}_S^i - \mathbf{C}_T) \right)^j \quad (1.58)$$

for the source domain features, and with

$$\frac{\partial \mathcal{L}_{\text{CORAL}}}{\partial f(\mathbf{U}_T^i)^j} = \frac{1}{d^2(n_T - 1)} \left((f(\mathbf{U}_T^i)^T - \frac{1}{n_T} (\mathbf{1}_{n_T}^T f(\mathbf{U}_T^i))^T \mathbf{1}_{n_T}^T)^T (\mathbf{C}_S - \mathbf{C}_T^i) \right)^j \quad (1.59)$$

for the target domain features (Sun et al., 2016; Sun & Saenko, 2016). In the given expression, $f(\mathbf{U}_S^i)^j$ denotes the j -th dimension of the i -th source data, and similarly $f(\mathbf{U}_T^i)^j$ represents the j -th dimension of the i -th target data.

Despite the good results proposed by Sun et al. (2016) using Deep CORAL and its effectiveness, it should be noted that the covariance matrices \mathbf{C}_S and \mathbf{C}_T are symmetric positive definite⁸. Symmetric positive definite matrices of size $n \times n$ with $n \in \mathbb{N}$ do not form a subspace of Euclidean space but are instead a Riemannian manifold with non-positive curvature \mathcal{S}_{++}^n . Therefore, methods in \mathcal{S}_{++}^n that rely on the Euclidean metric are suboptimal, as the Euclidean metric fails to consider the inner curvature of the ambient space (Morero & Murino, 2017). To address the computational intensity of classical methods based on the Riemannian metric on \mathcal{S}_{++}^n , we motivate the use of two Bregman divergences that are computationally efficient. Specifically, we leverage the Jeffreys and Stein distances based on the Bregman divergence, as suggested by Harandi et al. (2014). To compute these divergences, we first define the Bregman matrix divergence and Burg divergence, and then apply them to derive the Jeffreys and Stein divergences. Note that \mathcal{S}_{++}^n refers to the space of $n \times n$ symmetric positive definite matrices, which is characterized by the condition that $\mathbf{C} \in \mathcal{S}_{++}^n$ if and only if $\mathbf{a}^T \mathbf{C} \mathbf{a} > 0, \forall \mathbf{a} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$.

Definition 1.6.3 (Bregman divergence, Harandi et al. (2014)). *Let \mathcal{S}_{++}^n be a symmetric positive cone and $\xi : \mathcal{S}_{++}^n \rightarrow \mathbb{R}$ be a strictly convex and differentiable function defined on \mathcal{S}_{++}^n . The Bregman matrix divergence $\text{Breg}_\xi : \mathcal{S}_{++}^n \times \mathcal{S}_{++}^n \rightarrow [0, \inf)$ is defined as*

$$\text{Breg}_\xi(\mathbf{C}_S, \mathbf{C}_T) = \xi(\mathbf{C}_S) - \xi(\mathbf{C}_T) - \text{Tr}((\nabla_{\mathbf{C}_T} \xi)^T (\mathbf{C}_S - \mathbf{C}_T)), \quad (1.60)$$

where $\nabla_{\mathbf{C}_T} \xi$ is the gradient of ξ with respect to \mathbf{C}_T . $\text{Tr}(\mathbf{C})$ is the trace of the matrix $\mathbf{C} \in \mathbf{R}^n$ of size n with $\text{Tr}(\mathbf{C}) = \sum_{i=1}^n c_{ii}$, where c_{ii} denotes the entry on the i -th row and the i -th column of the matrix \mathbf{C} . The Bregman divergence is non-negative and definite, and it holds:

$$\text{Breg}_\xi(\mathbf{C}_S, \mathbf{C}_T) = 0 \text{ iff } \mathbf{C}_S = \mathbf{C}_T. \quad (1.61)$$

⁸A symmetric matrix \mathbf{C} of size $n \times n$ is *positive definite* if the scalar $\mathbf{x}^T \mathbf{C} \mathbf{x} > 0$ is strictly positive for every non-zero column vector \mathbf{x} of $n \in \mathbb{N}$ real numbers (Trefethen & Bau, 1997).

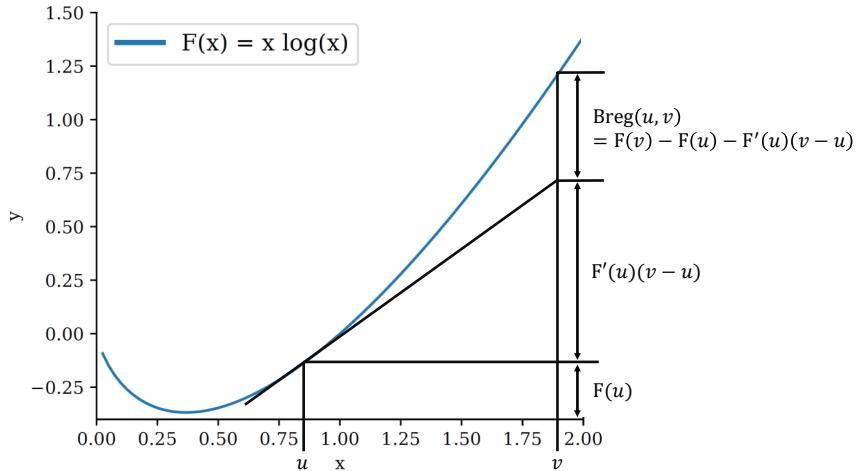


Figure 1.18: Visualization of the Bregman divergence. We assume a function $\text{Breg} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ and we define a strictly convex and differentiable function $F : \mathbb{R} \rightarrow \mathbb{R}$. Then, we can visualize the Bregman divergence $\text{Breg}(\cdot, \cdot)$ (Sun, 2010; Cichocki et al., 2015).

Figure 1.18 visualizes the Bregman divergence.

Definition 1.6.4 (Euclidean distance, Harandi et al. (2014)). *The Euclidean distance (i.e., the Frobenius norm for matrices) is obtained by using $\xi(\mathbf{C}) = \text{Tr}(\mathbf{C}^T \mathbf{C})$ as seed function in the Bregman divergence.*

For CORAL in Equation 1.55 we use the Frobenius distance in Definition 1.6.3. The Jeffreys and Stein divergences are based on the Burg divergence defined in the following.

Definition 1.6.5 (Burg divergence, Harandi et al. (2014)). *The Burg divergence, often referred to as B-divergence, is obtained by $\xi(\mathbf{C}) = -\log \det(\mathbf{C})$ as seed function in the Bregman divergence in Definition 1.6.3. $\det(\mathbf{C})$ denotes the determinant⁹ of the matrix \mathbf{C} . Then, the Burg divergence is defined as*

$$\text{Burg}(\mathbf{C}_S, \mathbf{C}_T) = \text{Tr}(\mathbf{C}_S \mathbf{C}_T^{-1}) - \log \det(\mathbf{C}_S \mathbf{C}_T^{-1}) - n. \quad (1.62)$$

As a result of the asymmetric behavior exhibited by the Bregman divergence, it is often unsuitable for use in real-world scenarios (Cherian et al., 2012). Thus, we proceed to establish the symmetrized Bregman divergences, referred to as the Jeffreys and Stein divergences, in the following.

Definition 1.6.6 (Jeffreys divergence, Harandi et al. (2014)). *The Jeffreys divergence,*

⁹The determinant $\det(\mathbf{C})$ of an $n \times n$ matrix \mathbf{C} can be computed with the Leibniz formula.

often referred to as J -divergence, is defined as

$$\begin{aligned}
J(\mathbf{C}_S, \mathbf{C}_T) &= \frac{1}{2} \text{Burg}(\mathbf{C}_S, \mathbf{C}_T) + \frac{1}{2} \text{Burg}(\mathbf{C}_T, \mathbf{C}_S) \\
&= \frac{1}{2} \text{Tr}(\mathbf{C}_S \mathbf{C}_T^{-1}) - \frac{1}{2} \log \det(\mathbf{C}_S \mathbf{C}_T^{-1}) + \frac{1}{2} \text{Tr}(\mathbf{C}_T \mathbf{C}_S^{-1}) - \frac{1}{2} \log \det(\mathbf{C}_T \mathbf{C}_S^{-1}) - n \\
&= \frac{1}{2} \text{Tr}(\mathbf{C}_S \mathbf{C}_T^{-1}) + \frac{1}{2} \text{Tr}(\mathbf{C}_T \mathbf{C}_S^{-1}) - n.
\end{aligned} \tag{1.63}$$

Definition 1.6.7 (Stein divergence, Sra (2012); Harandi et al. (2014)). *The Stein divergence, often referred to as S -divergence, is defined as*

$$\begin{aligned}
S(\mathbf{C}_S, \mathbf{C}_T) &= \frac{1}{2} \text{Burg}\left(\mathbf{C}_S, \frac{\mathbf{C}_S + \mathbf{C}_T}{2}\right) + \frac{1}{2} \text{Burg}\left(\mathbf{C}_T, \frac{\mathbf{C}_S + \mathbf{C}_T}{2}\right) \\
&= \log \det\left(\frac{\mathbf{C}_S + \mathbf{C}_T}{2}\right) - \frac{1}{2} \log \det(\mathbf{C}_S \mathbf{C}_T).
\end{aligned} \tag{1.64}$$

Due to the violation of the triangular inequality by $J(\mathbf{C}_S, \mathbf{C}_T)$, as shown in Lemma 1.6.1, the Jeffreys divergence-based distance is not a proper distance. Nevertheless, it is commonly used (Morerio & Murino, 2017). Since $J(\mathbf{C}_S, \mathbf{C}_T)$ and $S(\mathbf{C}_S, \mathbf{C}_T)$ are defined in terms of matrix multiplications, these divergences exhibit poor scaling with respect to the dimension n of \mathbf{C}_S and \mathbf{C}_T . In addition, it should be noted that computing $S(\mathbf{C}_S, \mathbf{C}_T)$ involves the computation of a matrix inverse. On the other hand, the Burg divergence is designed to be invariant to affine transformation, as it satisfies $\text{Burg}(\mathbf{C}_S, \mathbf{C}_T) = \text{Burg}(\mathbf{Q} \mathbf{C}_S \mathbf{Q}^T, \mathbf{Q} \mathbf{C}_T \mathbf{Q}^T)$ for any $\mathbf{Q} \in \text{GL}(n)$, where $\text{GL}(n)$ denotes the general linear group of real invertible $n \times n$ matrices. Since the Definition 1.6.6 of the Jeffreys divergence and the Definition 1.6.7 of the Stein divergence are derived from the Definition 1.6.5 of the Burg divergence, the Jeffreys and Stein divergences also exhibit affine invariance. This property is particularly important for computer vision algorithms. In contrast, the Euclidean distance in Definition 1.6.4 is only invariant to rotation. (Cherian et al., 2012; Sra, 2012; Harandi et al., 2014)

From the Definition 1.6.6 of the Jeffreys divergence, we can reformulate Equation 1.55 of the standard Deep CORAL to the Jeffreys CORAL loss function

$$\mathcal{L}_{\text{CORAL,Jeffreys}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \frac{1}{4d^2} \left(\frac{1}{2} \text{Tr}(\mathbf{C}_S \mathbf{C}_T^{-1} + \mathbf{C}_T \mathbf{C}_S^{-1}) - n \right), \tag{1.65}$$

and respectively with the Definition 1.6.7 of the Stein divergence to the Stein CORAL loss function

$$\mathcal{L}_{\text{CORAL,Stein}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \frac{1}{4d^2} \left(\log \det\left(\frac{1}{2} \mathbf{C}_S + \frac{1}{2} \mathbf{C}_T\right) - \frac{1}{2} \log \det(\mathbf{C}_S \mathbf{C}_T) \right), \tag{1.66}$$

where \mathbf{C}_S is the covariance of the source domain feature embeddings $f(\mathbf{U}_S)$, and \mathbf{C}_T is the covariance of the target domain feature embeddings $f(\mathbf{U}_T)$.

Application in Research. The use of Bregman divergences has proven to be significant in various domains, including computer vision classification (Harandi et al., 2014), as well as in face and vehicle tracking and texture segmentation (Cherian et al., 2012). For instance, CORAL has been employed for time-series DA in human activity recognition (Shi et al., 2022). Deep CORAL has been implemented in the AdaTime toolbox (Ragab et al., 2023) as a DA loss for time-series classification in activity recognition from sensors (such as accelerometer, gyroscope, body sensors, or electroencephalography signals). The effectiveness of Deep CORAL has been evaluated on medical time-series data (Ozyurt et al., 2023). Therefore, (Deep) CORAL has become a widely adopted metric for feature alignment across diverse domains.

Application in Contributing Papers. In our previous work (Ott et al., 2022a), we employed the solution of the standard CORAL, as defined in Theorem 1.6.2, by utilizing the SVD to obtain the transformation matrix from the source domain to target domain. To compare the second-order distances between feature embeddings of source and target samples, we utilized the standard CORAL metric (Equation 1.55), along with the Jeffreys CORAL metric (Equation 1.65), and the Stein CORAL metric (Equation 1.66). In another work (Ott et al., 2022b), we adapted feature embeddings of sensor data captured by a digital pen for writing on paper to those for writing on tablet. To accomplish this, we utilized the differentiable loss functions of the standard, Jeffreys, and Stein CORAL to train two parallel network pipelines. We also evaluated five different layers as DA points. Moreover, we extended our work by incorporating Deep CORAL, which is implemented in the AdaTime toolbox (as discussed in Chapter 2), with the Jeffreys and Stein CORAL loss functions. We further evaluated a combination of Sinkhorn with all three CORAL variants.

1.6.7 Higher-Order Moment Matching

The majority of current techniques aim to match the second-order or lower moments, which may pose limitations in terms of statistical characteristics for non-Gaussian distributions. To address this issue, higher-order moment matching (HoMM) of order ≥ 3 has been proposed by Chen et al. (2020) to perform domain alignment by considering higher-order statistics and address more intricate distributions. HoMM is defined by

$$\mathcal{L}_{\text{HoMM}_{p,H,bs}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \frac{1}{H^p} \left\| \frac{1}{n_S} \sum_{i=1}^{n_S} f(\mathbf{U}_S^i)^{\otimes p} - \frac{1}{n_T} \sum_{i=1}^{n_T} f(\mathbf{U}_T^i)^{\otimes p} \right\|_F^2, \quad (1.67)$$

between the embedding $f(\mathbf{U}_S)$ of the source domain and the embedding $f(\mathbf{U}_T)$ of the target domain, with $f(\mathbf{U}_{(\cdot)}^i) = [f(\mathbf{U}_{(\cdot)}^i)(1), f(\mathbf{U}_{(\cdot)}^i)(2), \dots, f(\mathbf{U}_{(\cdot)}^i)(H)] \in \mathbb{R}^H$. It holds $n_S = n_T = bs$, with the batch size bs . $\|\cdot\|_F$ denotes the Frobenius norm, H is the number of hidden neurons in the adapted layer, and $(\cdot)^{\otimes p}$ denotes the p -level tensor power. In the case of $p = 1$, HoMM is identical to linear MMD (Tzeng et al., 2014) (refer to Section 1.6.4),

while for $p = 2$, it is equivalent to Gram matrix matching. The centralized Gram matrix is transformed into a covariance matrix when activation outputs are normalized (i.e., by subtracting the mean value). Consequently, the second-order HoMM ($p = 2$) is equivalent to the CORAL (Sun & Saenko, 2015; Sun et al., 2016) method (see Section 1.6.6). However, since HoMM can perform moment matching for higher orders ($p \geq 3$), the space complexity for calculating the tensor $(\cdot)^{\otimes p}$ increases to $\mathcal{O}(H^p)$, which becomes infeasible for our models' embedding sizes ranging from 60×100 to 400×200 . For instance, for $H = 200$, the complexity becomes $\mathcal{O}(10^6)$ for $p = 3$, and $\mathcal{O}(10^9)$ for $p = 4$. As a solution, Chen et al. (2020) proposed group moment matching.

Group moment matching (GMM). Let the division of hidden neurons into n_g groups be denoted by $\mathcal{G} = \{\mathbf{G}_1, \mathbf{G}_2, \dots, \mathbf{G}_{n_g}\}$, where each group \mathbf{G}_i has $\lfloor \frac{H}{n_g} \rfloor$ neurons. The corresponding HoMM loss function with GMM is defined as:

$$\mathcal{L}_{\text{HoMM}, \text{GMM}_{p, H, bs, n_g}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \frac{1}{bs^2 \lfloor \frac{H}{n_g} \rfloor^p} \sum_{j=1}^{n_g} \left\| \sum_{i=1}^{bs} f(\mathbf{U}_S^{i,j})^{\otimes p} - \sum_{i=1}^{bs} f(\mathbf{U}_T^{i,j})^{\otimes p} \right\|_F^2, \quad (1.68)$$

where $f(\mathbf{U}_S^{i,j})$ and $f(\mathbf{U}_T^{i,j})$ are the activation outputs of the j -th group. The space complexity decreases to $\mathcal{O}(n_g \cdot \lfloor \frac{H}{n_g} \rfloor^p)$ (Chen et al., 2020). However, for $p \geq 5$, group moment matching still does not work effectively for practical applications and higher layer sizes.

Random sampling matching (RSM). The process of HoMM with RSM can be expressed formally as follows: First, a high-level tensor is considered, from which RSM randomly selects T values. Then, these T values are aligned in both feature embeddings. Through random sampling matching, HoMM is capable of performing arbitrary-order moment matching. Hence, HoMM with RSM can be defined by

$$\mathcal{L}_{\text{HoMM}, \text{RSM}_{p, T, bs}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \frac{1}{bs^2 T} \sum_{j=1}^T \left[\sum_{i=1}^{bs} \prod_{k=r[j,1]}^{r[j,p]} [f(\mathbf{U}_S^i)](k) - \sum_{i=1}^{bs} \prod_{k=r[j,1]}^{r[j,p]} [f(\mathbf{U}_T^i)](k) \right]^2. \quad (1.69)$$

$r \in \mathbb{R}^{T \times p}$ denotes the randomly generated position index matrix. Then, the randomly sample value in the p -level tensor $f(\mathbf{U}_{(\cdot)}^i)^{\otimes p}$ is denoted by $\prod_{k=r[j,1]}^{r[j,p]} [f(\mathbf{U}_{(\cdot)}^i)](k)$. The space complexity reduces to $\mathcal{O}(T)$ (Chen et al., 2020).

Kernelized HoMM. Similar to the kernelization of MMD (Long et al., 2017) in Section 1.6.4, HoMM can be kernelized by a reproducing kernel Hilbert space (RKHS). Based on Equation 1.67, kernelized HoMM (kHoMM) becomes

$$\mathcal{L}_{\text{kHoMM}_{p, H, bs}}(f(\mathbf{U}_S), f(\mathbf{U}_T)) = \frac{1}{H^p} \left\| \frac{1}{bs} \sum_{i=1}^{bs} \phi(f(\mathbf{U}_S^i)^{\otimes p}) - \frac{1}{bs} \sum_{i=1}^{bs} \phi(f(\mathbf{U}_T^i)^{\otimes p}) \right\|_F^2, \quad (1.70)$$

where $\phi(f(\mathbf{U}_{(\cdot)}^i)^{\otimes p})$ is the feature's i -th sample in RKHS (Chen et al., 2020). The kHoMM algorithm can be formulated similarly to the HoMM algorithm with random sampling matching, as shown in Equation 1.69, where kHoMM can be randomly sampled from a radial basis function (RBF) kernel. When $p = 1$, kHoMM becomes equivalent to kMMD as proposed by Long et al. (2017).

Application in Research. The effectiveness of HoMM and kHoMM for domain adaptation in optical digits recognition datasets and optical object recognition was assessed by Chen et al. (2020). On the other hand, the performance of HoMM was evaluated in MTS activity recognition from sensors by AdaTime (Ragab et al., 2023), while Ozyurt et al. (2023) evaluated the effectiveness of HoMM on medical MTS data.

Application in Contributing Papers. In our previous work (Ott et al., 2022a), we utilize HoMM and kHoMM as a distance metric to select the most suitable transformation (computed with optimal transport) from the source to the target domain. In contrast, in Ott et al. (2022b), we employed HoMM and kHoMM as a loss function to minimize the discrepancy between feature embeddings at five different network layers. We compared HoMM and kHoMM of order $p = 2$ and $p = 3$ with random sampling matching and set $T = 1,000$ as the maximal limit for training, which was determined by the capabilities of the Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM. In Chapter 2, we utilized HoMM of order 3, similar to AdaTime, without group moment matching or random sampling matching, since the source and target domains were aligned at a late network layer with small layer sizes.

1.7 Dimensionality Reduction

In Section 1.6, it is noted that representation learning presents a challenge in determining its objective compared to classification tasks. The complexity and dimensionality of learned representations make it difficult to evaluate their quality, and achieving better results in classification does not necessarily indicate a good representation. Therefore, researchers resort to visualizing feature representations and analyzing the arrangement and overlapping of clusters to evaluate performance. However, as feature representations typically have high dimensionality (i.e., > 100), dimensionality reduction techniques such as principal component analysis (PCA) and t-stochastic neighbor embedding (t-SNE) are employed to visualize them in two or three dimensions. This section provides a brief summary of PCA and t-SNE, which are often used for visualization. Additionally, we describe the properties of t-SNE for feature visualization that were used in our contributing papers (Ott et al., 2022a,c, 2023c).

Principal Component Analysis (PCA). In accordance with Chan (2021), we provide a brief summary of the objective of PCA. PCA involves the eigendecomposition of the covariance matrix \mathbf{C} . Let $\mathcal{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n) \in \mathbb{R}^{n \times d}$ be a set of n data samples of d -dimensional vectors. PCA aims to find a low-dimensional representation in \mathbb{R}^p with $p \ll d$. We can define a set of basis vectors $\{\mathbf{v}_1, \dots, \mathbf{v}_p\}$, where each $\mathbf{v}_i \in \mathbb{R}^d$ approximates the input data by $\mathbf{x}_n \approx \sum_{i=1}^p \alpha_i \mathbf{v}_i$, with α as the representation coefficients. The basis vectors are optimally aligned with the geometric properties of the dataset, such that data samples have significant coefficients for major axes and small coefficients for minor axes. The reduction in dimensionality results in the preservation of only the larger coefficients. To compute

the basis vectors and the corresponding coefficients, we aim to minimize the optimization problem:

$$(\hat{\mathbf{v}}, \hat{\alpha}) = \arg \min_{\|\mathbf{v}\|_2=1, \alpha} \|\mathbf{x} - \alpha \mathbf{v}\|^2. \quad (1.71)$$

by taking the derivative with respect to α and set it to zero: $2\mathbf{v}^T(\mathbf{x} - \alpha \mathbf{v}) = 0 \Rightarrow \alpha = \mathbf{v}^T \mathbf{x}$. We set $\alpha = \mathbf{v}^T \mathbf{x}$ in the optimization problem in Equation 1.71 and get

$$\arg \min_{\|\mathbf{v}\|_2=1, \alpha} \|\mathbf{x} - \alpha \mathbf{v}\|^2 = \arg \max_{\|\mathbf{v}\|_2=1} \{\mathbf{v}^T \mathbf{x} \mathbf{x}^T \mathbf{v}\}, \quad (1.72)$$

which is independent of α . Assuming that \mathbf{x} is a realization of a random vector \mathbf{X} , the optimization problem of Equation 1.72 becomes

$$\arg \min_{\|\mathbf{v}\|_2=1, \alpha} \mathbb{E} \|\mathbf{X} - \alpha \mathbf{v}\|^2 = \arg \max_{\|\mathbf{v}\|_2=1} \mathbf{v}^T \mathbb{E}\{\mathbf{X} \mathbf{X}^T\} \mathbf{v} = \arg \max_{\|\mathbf{v}\|_2=1} \mathbf{v}^T \mathbf{C} \mathbf{v}, \quad (1.73)$$

where $\mathbf{C} := \mathbb{E}[\mathbf{X}^T \mathbf{X}]$ (Chan, 2021). This maximization is equivalent to the eigendecomposition.

Theorem 1.7.1 (Eigendecomposition, (Chan, 2021)). *The $d \times d$ -dimensional matrix \mathbf{C} has the eigendecomposition $\mathbf{C} = \mathbf{P} \boldsymbol{\Sigma} \mathbf{P}^T$, where \mathbf{P} is the $d \times d$ eigenvector matrix, and $\boldsymbol{\Sigma}$ the eigenvalue matrix. Then, the optimization $\hat{\mathbf{v}} = \arg \max_{\|\mathbf{v}\|_2=1} \mathbf{v}^T \mathbf{C} \mathbf{v}$ has a solution $\hat{\mathbf{v}} = \mathbf{p}_1$, the first column of the eigenvector matrix \mathbf{P} .*

For a proof of Theorem 1.7.1, refer to Chan (2021). To obtain further principal components of the covariance matrix, it is assumed that the eigenvectors in matrix $\boldsymbol{\Sigma}$ are arranged in descending order based on the magnitude of the corresponding eigenvalues.

t-SNE. The t-SNE algorithm, proposed by van der Maaten & Hinton (2008) as an extension of SNE (Hinton & Roweis, 2002), is used for the visualization of high-dimensional data samples $\mathcal{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n) \in \mathbb{R}^{n \times d_x}$ with d_x dimensions. The algorithm maps these samples to a lower-dimensional space, typically $d_y = 2$ or $d_y = 3$, denoted by $\mathcal{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n) \in \mathbb{R}^{n \times d_y}$, which can be visualized in a human-readable manner. t-SNE overcomes the issue of overcrowding of samples in the center of the map, and produces a map that captures the structure of data at different scales. The low-dimensional samples \mathbf{y}_i represent individual map points in \mathcal{Y} . The objective is for the map \mathcal{Y} to preserve a substantial portion of the essential structure present in the data \mathcal{X} . Although the linear method of PCA effectively maintains a considerable distance between the representations \mathcal{Y} of dissimilar data samples, it is not feasible for high-dimensional data that exist on a nonlinear manifold. A multitude of nonlinear dimensionality reduction techniques have been developed, such as locally linear embedding and stochastic neighbor embedding, but these techniques do not possess the capability to retain both the local and global structure of the data within a single map (van der Maaten & Hinton, 2008). Hence, we utilized t-SNE for both data and embedding visualization in the contributing papers (Ott et al., 2022a,c, 2023c), and provide a more detailed explanation of SNE and t-SNE in the following.

SNE computes the asymmetric probability $p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ik}^2)}$, that represents the probability that object i would pick object j , and the dissimilarities d_{ij}^2 are given by

$d_{ij}^2 = \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_i^2}$, where σ_i produces a P_i (i.e., the conditional probability distribution) with a fixed hyperparameter, the so-called *perplexity*, for P_i that is defined as $\text{Perp}(P_i) = 2^{S(P_i)}$, with the Shannon entropy $S(P_i) = -\sum_j p_{ij} \log p_{ij}$ of P_i (van der Maaten & Hinton, 2008). The probability in the low-dimensional space is

$$q_{ij} = \frac{\exp(-\|\mathbf{y}_i - \mathbf{y}_j\|^2)}{\sum_{i \neq k} \exp(-\|\mathbf{y}_i - \mathbf{y}_k\|^2)}, \quad (1.74)$$

also with Gaussian neighborhoods (with $\sigma = \frac{1}{2}$). By minimizing the PC divergence as cost function with, for instance, gradient descent, both distributions p_{ij} and q_{ij} can be matched, where P_i and Q_i represent the conditional probability distribution over all samples (in high-dimensional space, respectively in low-dimensional space) (Hinton & Roweis, 2002). The gradient of the cost function c is

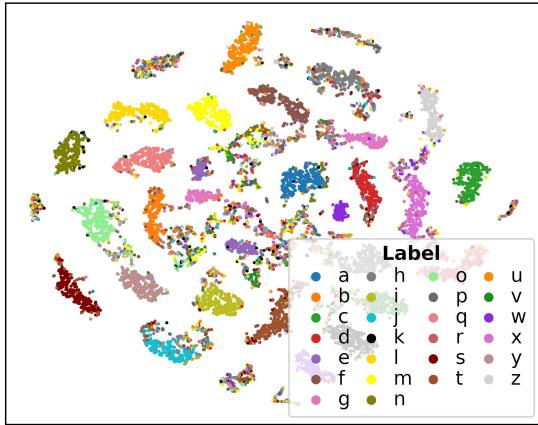
$$\frac{\partial c}{\partial \mathbf{y}_i} = 2 \sum_j (p_{ij} - q_{ij} + p_{ji} - q_{ji})(\mathbf{y}_i - \mathbf{y}_j). \quad (1.75)$$

Then, the gradient update is given by

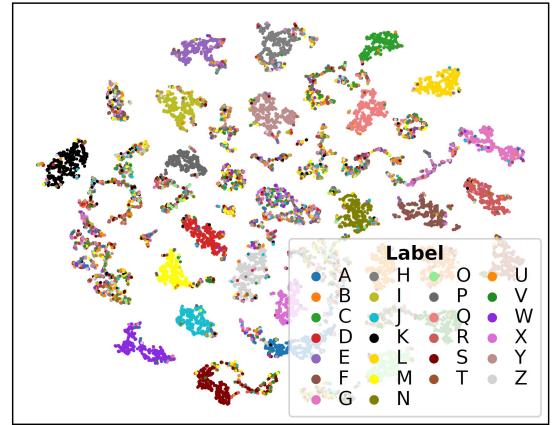
$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\partial c}{\partial \mathcal{Y}} + \alpha^{(t)} (\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}), \quad (1.76)$$

at iteration t with learning rate η and the momentum $\alpha^{(t)}$. As the momentum and the step size has to be chosen carefully, a hyperparameter search should be performed. In contrast to SNE, t-SNE uses a symmetrized cost function and a Student-t distribution rather than a Gaussian distribution in Equation 1.74 (van der Maaten & Hinton, 2008).

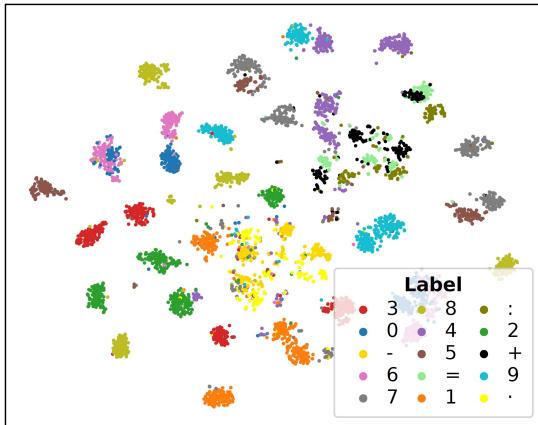
Time-Series Embedding Visualization. In our experiments with various datasets, we established distinct hyperparameters and refrained from selectively choosing embedding visualizations. The initial dimension ranges from 1,500 (formed by reshaping 50×30) and 3,800 (formed by reshaping 19×200). Initially, we utilized PCA to decrease the dimensionality of the data to the specified *initial_dimension*. Subsequently, we employed t-SNE with the *perplexity* parameter to obtain a two-dimensional representation. We iterate t-SNE for 2,000 time steps, initialized the momentum to 0.5, and then increased it to 0.8 after 20 iterations. We exemplarily present visualizations of time-series representations of handwritten samples obtained from a sensor-enhanced pen (refer to Figure 1.19) and generated sinusoidal time-series samples with induced noise (refer to Figure 1.20). In Figure 1.19a and Figure 1.19b, samples belonging to different classes are clustered together, resulting in misclassification. Additionally, in Figure 1.19c, clusters representing samples labeled as '+' (black), '=' (light green), and ':' (brown) overlap, leading to lower classification accuracy. In Figure 1.19d, the clusters representing right-handed (blue) and left-handed (orange) writers are separated in the two-dimensional representation. However, due to domain shift between the two types of writers, a combined training of a model results in reduced accuracy. This domain shift can be mitigated by transforming the representations of left-handed writers to align with those of right-handed writers, resulting in



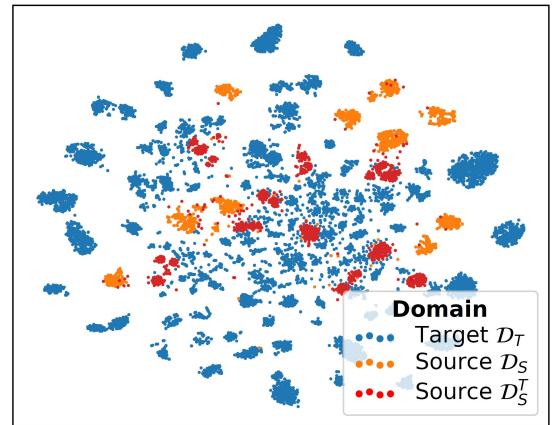
(a) Visualization of lowercase character embeddings.



(b) Visualization of uppercase character embeddings.



(c) Visualization of number and symbol embeddings.

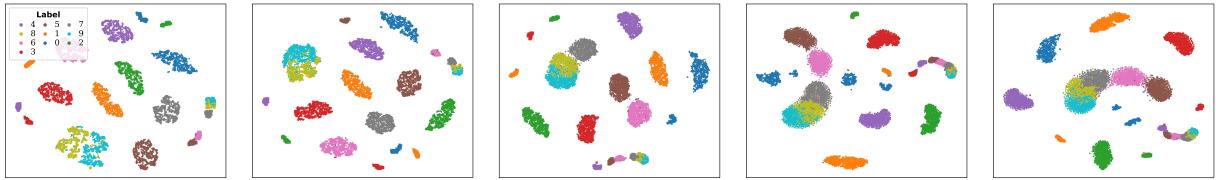


(d) Visualization of character embeddings recorded from right-handed writers (blue), left-handed writers (orange), and transformed left-handed writers (red).

Figure 1.19: Two-dimensional visualization of embeddings of OnHW time-series data using t-SNE with PCA as initial dimensionality reduction. Source: Ott et al. (2022a).

closer representations in the low-dimensional space. In the generated sinusoidal dataset, clusters are clearly distinguishable, but the representations of class labels '8' (light green) and '9' (light blue) overlap (refer to Figure 1.20a). With an increase in noise, the clusters of representations become closer, as seen in the five clusters in the middle of Figure 1.20e.

Discussion About t-SNE Visualizations. It is essential to analyze t-SNE visualizations carefully since they can be misleading. Choosing the right perplexity value is crucial as it balances the local and global aspects of the dataset. High perplexity values lead to



(a) Sinusoidal em- (b) Sinusoidal em- (c) Sinusoidal em- (d) Sinusoidal (e) Sinusoidal
beddings without beddings with low beddings with in- embeddings with embeddings with
noise. noise. intermediate noise. higher noise. highest noise.

Figure 1.20: Two-dimensional visualization of embeddings of generated sinusoidal time-series data with the classes 0 to 9 using t-SNE with PCA as initial dimensionality reduction. Source: Ott et al. (2022a).

cluster merging, resulting in a single big cluster, while low perplexity values lead to the presence of many small, closely spaced clusters. Denser datasets require higher perplexity values. A significant drawback of t-SNE is its high computational time, as demonstrated by the $\approx 10\text{ h}$ required to compute the representation in Figure 1.19. Moreover, t-SNE can get trapped in local optima. One significant advantage of t-SNE is its correlation to some extent with the performance of a classification model. When samples are distinctly separated in the low-dimensional representation, the high-dimensional samples contain adequate variability to build a classifier with good classification accuracy. Regarding the number of iterations, it is essential to continue iterating until a stable configuration is reached. Furthermore, if high-dimensional clusters have different standard deviations, the low-dimensional clusters are approximately the same size, as t-SNE expands dense clusters and contracts sparse clusters. Consequently, relative distance in the low-dimensional representation are meaningless, while distances between clusters depend on the perplexity parameter.

Chapter 2

Practical Applications of Domain Adaptation for Time-Series Classification

Over the past few years, a diverse range of DA methods have been developed for time-series classification, but a comprehensive assessment of these methods across various datasets is lacking. This chapter aims to establish a benchmark of DA methods using the AdaTime toolbox (see Section 2.1). The techniques for feature alignment are summarized in Section 2.2. Furthermore, we present details regarding datasets such as sleep, gesture, epilepsy, and human activity recognition, online handwriting recognition, interference detection based on global navigation satellite system (GNSS) data, and generated sinusoidal data, as well as dataset challenges and information about decoder networks in Section 2.3. The evaluation results and hyperparameter searches are proposed in Section 2.4.

2.1 Benchmarking Toolbox & Contributions

Evaluation schemes in existing works for DA lack consistency, as comparing methods using different network encoders can result in improper comparability. Additionally, there is currently no standard for evaluation datasets in the field of time-series classification. To address these issues in our evaluation, we utilize the benchmark suite AdaTime¹⁰ proposed by Ragab et al. (2023). This suit standardizes the neural network encoders, implements ten methods for unsupervised or few-shot DA learning, and evaluates them on four sleep and human activity datasets. The authors demonstrate that visual DA methods can be adapted to time-series classification tasks and are competitive when coupled with careful hyperparameter searches. The procedure of AdaTime involves several steps. Firstly, the dataset is prepared by slicing, splitting, and normalization. Next, an encoder network extracts the features from the source and target domains. An unsupervised DA method is then utilized to mitigate the domain shift between the extracted features of the two

¹⁰Github AdaTime: <https://github.com/emadeldeen24/AdaTime>

domains. To facilitate fair hyperparameter selection without using training labels (in the unsupervised setup) or with only a few training labels (in the few-shot learning setup), Ragab et al. (2023) implemented source risk (Ganin et al., 2016), validation risk (You et al., 2019), target risk, and few-shot target risk. In this section, we utilize the notations introduced in Section 1.6. Specifically, we define the source domain data $\mathcal{D}_S = \{\mathcal{U}_S^i, \mathcal{Y}_S^i\}_{i=1}^{n_S}$, consisting of $n_S \in \mathbb{N}$ labeled samples with MTS source data \mathcal{U}_S^i . Similarly, we define the target domain data as $\mathcal{D}_T = \{\mathcal{U}_T^i, \mathcal{Y}_T^i\}_{i=1}^{n_T}$, consisting of $n_T \in \mathbb{N}$ (un)labeled samples with MTS target data \mathcal{U}_T^i . We assume that both domains share the same label space $\mathcal{Y} = \mathcal{Y}_S = \mathcal{Y}_T$, where $\mathcal{Y} = \{1, \dots, |\mathcal{Y}|\}$ and $|\mathcal{Y}|$ are the number of classes. The goal of DA methods is to align representations of the source and target domains by finding domain-invariant features. This is achieved by minimizing the distance between the features of the source domain $f(\mathcal{U}_S)$ and the target domain $f(\mathcal{U}_T)$. This domain alignment loss is given by

$$\mathcal{L}_{\text{align}} = \min_f d(f(\mathcal{U}_S), f(\mathcal{U}_T)), \quad (2.1)$$

where d is a distance metric (see Section 1.6). Furthermore, the neural network maps the extracted and aligned features to their corresponding class probabilities, which are typically computed using the cross-entropy loss function \mathcal{L}_{CE} . In AdaTime, and typically in most DA methods, the alignment and classification loss functions are jointly optimized, which can be expressed by

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{align}} + \mathcal{L}_{\text{CE}}. \quad (2.2)$$

By jointly optimizing the alignment and classification loss functions, the domain shift is mitigated, while simultaneously learning the source classification task (Ragab et al., 2023).

We present our contributions as follows: (1) We utilize the AdaTime (Ragab et al., 2023) toolbox as a baseline to benchmark DA for time-series classification. Our primary focus is on evaluating DML and representation learning techniques, such as feature alignment loss functions and Sinkhorn transport. To achieve this goal, we use ten DA methods implemented by (Ragab et al., 2023) and extend the toolbox with 14 additional representation learning methods. The objective is to compare our results with those obtained using the optimal transport method proposed by Ott et al. (2022a), particularly on the OnHW datasets and sinusoidal dataset. Moreover, we assess the effectiveness of combining Sinkhorn with other metric learning functions. (2) We extend the evaluation of DA methods to a broader range of datasets than those used in Ragab et al. (2023). Specifically, we evaluate on three gesture recognition datasets, epilepsy and face recognition datasets, an OnHW dataset for writing on tablet, three single character-based OnHW datasets (i.e., OnHW-chars, OnHW-symbols, and split OnHW-equations) for writing on paper, a multivariate and binary GNSS-based dataset for interference detection, i.e., time-series anomaly detection (Goswami et al., 2023), and a sinusoidal generated univariate time-series dataset. In contrast to previous work (Wilson et al., 2020; Liu & Xe, 2021; He et al., 2023; Ragab et al., 2023), we train every possible combination of source and target domain data scenario, including between participants of data collection, between left-handed and right-handed writers and vice versa, between channels of sensor data, and between sinus and cosine data. This approach allows for more generalizable statements. Overall, we train 1,405,290

domain scenario combinations. (3) We follow the AdaTime approach of visualizing dataset statistics, such as class distributions, to highlight the challenges of DA. (4) In addition to evaluating the performance of DA methods, we conduct a detailed analysis of the source, target, and few-shot target risks, and (5) to further improve the results of DA, we propose an extensive hyperparameter search of the CNN, ResNet, and TCN encoder networks implemented by Ragab et al. (2023).

2.2 Domain Adaptation Methods

This section provides an overview of the DA methods used for time-series classification tasks, including those implemented in AdaTime as well as additional methods used for benchmarking. For a categorization of DA techniques, please refer to Section 1.5.3. While some methods were originally designed for visual DA applications (Tzeng et al., 2014; Sun et al., 2016; Ganin et al., 2016; Long et al., 2018; Shu et al., 2018; Rahman et al., 2020; Chen et al., 2020; Zhu et al., 2020b), three were specifically developed for time-series DA (Wilson et al., 2020; Liu & Xe, 2021; Ott et al., 2022a). All of the methods can be adapted for time-series classification. Discrepancy-based methods (Tzeng et al., 2014; Sun et al., 2016; Chen et al., 2020; Rahman et al., 2020; Zhu et al., 2020b; Ott et al., 2022a) aim to minimize the distance between source and target domains, while adversarial-based methods (Ganin et al., 2016; Long et al., 2018; Shu et al., 2018; Wilson et al., 2020; Liu & Xe, 2021) use a domain discriminator network. Marginal techniques (Tzeng et al., 2014; Ganin et al., 2016; Sun et al., 2016; Chen et al., 2020; Wilson et al., 2020; Liu & Xe, 2021; Ott et al., 2022a) align the marginal distribution, while other methods (Long et al., 2018; Shu et al., 2018; Rahman et al., 2020; Zhu et al., 2020b) jointly align both marginal and conditional distributions. In the following, we give an overview of 24 unsupervised DA methods and the supervised DA method by Ott et al. (2022a).

MSE. We utilize the Frobenius norm to calculate the distance between feature embeddings using the standard mean squared error (MSE), as discussed in Section 1.6.1.

CS & PC. The Cosine similarity (CS) and Pearson correlation (PC) (Pearson, 1920) metrics measure the orthogonality or decorrelation between two vectors. In our work, we use them to increase the similarity between feature embeddings and align them across domains, as discussed in Section 1.6.2.

Kullback-Leibler & Jensen-Shannon Divergence. The Kullback-Leibler (KL) divergence, introduced by Kullback & Leibler (1951), and Jensen-Shannon divergence (JSD) are statistical measures that quantify the difference between two probability distributions by measuring the relative entropy or information divergence (see Section 1.6.3).

Linear MMD & kMMD. We apply the maximum mean discrepancy (MMD) to align the first order statistics (see Section 1.6.4). We assess both the linear MMD (Borgwardt

et al., 2006) and the kernelized MMD (kMMD) (Long et al., 2017) using a Gaussian RBF kernel.

Deep CORAL. The deep correlation alignment (Deep CORAL) method (Sun et al., 2016) is a technique to align the second-order statistics of source and target embeddings. It uses a differentiable loss function to accomplish this. For further details, please refer to Section 1.6.6.

Jeffreys & Stein CORAL. The affine invariant variations of the CORAL method, Jeffreys and Stein CORAL (Cherian et al., 2012; Sra, 2012; Harandi et al., 2014), are designed to align second-order statistics of source and target embeddings. Definition 1.6.6 and Definition 1.6.7 in Section 1.6.6 provide more information about these variations. We implement Equation 1.65 and Equation 1.66 to train the loss functions for the DA task.

Linear & Kernelized MMCD. Maximum mean and covariance discrepancy (MMCD) (Zhang et al., 2020a; Alipour & Tahmoresnezhad, 2021) is a combination of MMD and CORAL loss functions that aligns both the first and second-order statistics (see Section 1.6.5, Equation 1.47). We refer to the linear and kernelized MMCD, depending on whether the linear MMD or kMMD is used.

MMDA. The minimum discrepancy estimation for deep domain adaptation (MMDA) method (Rahman et al., 2020) leverages a conditional entropy minimization technique to integrate the kMMD and Deep CORAL alignment functions (refer to Section 1.6.4 and Section 1.6.6 for more information).

DDC. Deep domain confusion (DDC) (Tzeng et al., 2014) aims to minimize the distance between domains by using the kMMD metric, as explained in Section 1.6.4.

Linear & Squared DAN. The deep adaptation network (DAN) method, proposed by Long et al. (2015), embeds the hidden representations of all task-specific layers in an RKHS to align the mean embeddings of different domain distributions using multiple kMMD. We assess the performance of two version of loss functions, namely linear and squared DAN.

HoMM. The higher-order moment matching (HoMM) (Chen et al., 2020) approach aims to align domains based on higher-order statistics by minimizing the discrepancy between different domains. We specifically evaluate the HoMM method (without GMM and RSM) with order $p = 3$. Further details can be found in Section 1.6.7.

DANN. A gradient reversal layer is employed in the domain-adversarial neural network (DANN) (Ganin et al., 2016) to encourage the emergence of features that are discriminative for the source domain but indiscriminative with respect to the shift between the source and target domains.

CDAN. The conditional domain adversarial network (CDAN) (Long et al., 2018) learns representations that are disentangled and transferable by conditioning the adversarial adaptation models on discriminative information. The multi-linear conditioning captures the cross-covariance between the feature representation and classifier predictions, while the entropy conditioning controls the uncertainty of the classifier predictions.

DIRT-T. The decision-boundary iterative refinement training with a teacher (DIRT-T) (Shu et al., 2018) approach employs a combination of domain adversarial training and a penalty term that penalizes violations of the cluster assumption. Additionally, it utilizes natural gradient to further minimize the cluster assumption violation.

DSAN. A transfer network is learned by deep subdomain adaptation network (DSAN) (Zhu et al., 2020b), which aligns subdomain distributions of domain-specific layers across different domains based on a local kMMD distance.

CoDATS. The convolutional deep domain adaptation model for time-series data (CoDATS) technique, as proposed by Wilson et al. (2020), suggests a method for time-series DA through the use of a weak supervision DA method in the form of target-domain label distributions.

AdvSKM. Adversarial spectral kernel matching (AdvSKM) (Liu & Xe, 2021) aims to improve the MMD metric through the use of a hybrid spectral kernel (i.e., kMMD). This modified metric allows for a more accurate characterization of non-stationary and non-monotonic statistics within time-series distributions.

Sinkhorn. The concept of geometric distances involves measuring the optimal transportation cost, which can be calculated by solving a linear program, as originally proposed by Kantorovitch (2006). This approach has been demonstrated to be effective in various applications. An iterative method commonly used is the Sinkhorn¹¹ transport method, as presented by Courty et al. (2016). Further details can be found in Section 1.5.4.

Optimal Transport. The supervised DA method utilizing optimal transport (OT) (Ott et al., 2022a) aims to minimize the distance between the source and target domain features for each class label by employing the Earth Mover’s distance (EMD) and Sinkhorn transport techniques. The optimal transformation is then selected based on the alignment of the features, which is evaluated using CS and PC (see Section 1.6.2), MMD (see Section 1.6.4), kMMD (see Section 1.6.4), standard, Jeffreys, and Stein CORAL (see Section 1.6.6), and HoMM (without GMM and RSM) with order $p = 3$ (see Section 1.6.7).

¹¹Github Sinkhorn OT: <https://gist.github.com/wohlert/8589045ab544082560cc5f8915cc90bd>

Table 2.1: Statistics about time-series datasets with domain shift.

Dataset	# Domains	# Channels	# Classes	Length	Train Set	Test Set
HHAR ¹	9	3	6	128	12,716	5,218
UCI HAR ¹	30	9	6	128	2,300	990
WISDM ¹	36	3	6	128	1,350	720
SSC (EEG) ¹	20	1	5	3,000	14,280	6,130
uWave ²	8	3	5	150	3,582	896
Finger movements ²	28	1	2	50	104	104
Gestures mid air ²	10	66	14	66	1,323	1,326
Epilepsy ²	2	3	4	206	137	138
Face detection ²	2	144	2	62	6,966	2,448
PenDigits ²	2	2	10	8	8,133	2,859
OnHW-symbols	2	13	15	79	2,142	525
Split OnHW-equations	2	13	15	79	36,718	9205
OnHW-chars (combined)	2	13	52	64	24,887	8,658
GNSS (binary)	2	88	2	4 – 29	∅ 15,403	∅ 6,604
GNSS (multi-class)	2	88	9	4 – 29	∅ 15,403	∅ 6,604
Sinusoidal data	2	1	10	1,000	11,000	1,000

¹Dataset available at: <https://researchdata.ntu.edu.sg/dataverse/adatime>

²Dataset available at: <http://www.timeseriesclassification.com/dataset.php>

Sinkhorn + Alignment Loss. The MMD and optimal transport distances are designed to consider the underlying geometry of the underlying space in which the data resides. Feydy et al. (2019) demonstrated that the Sinkhorn divergence provides a means to interpolate between MMD and OT distances. Given the practical advantages of Sinkhorn, we explore its use in combination with alignment loss functions such as linear MMD, kMMD, Deep CORAL, Jeffreys CORAL, Stein CORAL, and HoMM. This combination of two geometric functions enables us to examine their impact on the alignment of feature representations.

2.3 Experimental Setup

We provide an overview of the time-series classification datasets used in our experimental setup and provide a rationale for the DA task (see Section 2.3.1). Subsequently, in Section 2.3.2, we detail the encoder architectures and their corresponding hyperparameters.

2.3.1 Datasets

In this section, we present various time-series datasets that serve for our comprehensive evaluation benchmark. Table 2.1 provides an overview of statistics associated with 15 such datasets, including the number of domains (i.e., participants involved in data collections), number of channels (i.e., univariate or multivariate data), number of classes, length of the time-series, and size of the training and test datasets. The datasets are summarized below.

HHAR. The dataset used for heterogeneity human activity recognition (HHAR) (Stisen et al., 2015) consists of human activities performed by nine users. The dataset includes readings obtained from three-axis accelerometers, with sampling rates ranging between 25 Hz and 200 Hz. The HHAR dataset was recorded using a total of 36 smartphones, tablets, and smartwatches, in order to analyze heterogeneities specific to sensors, devices, and workloads, since impairments can vary significantly across devices and are dependent on the type of recognition technique used. We evaluate all 36 possible combinations of DA between the nine users, while previous studies such as Wilson et al. (2020); Liu & Xe (2021); He et al. (2023); Ragab et al. (2023) evaluated only five combinations.

UCI HAR. The UCI human activity recognition (HAR) (Anguita et al., 2013) dataset comprises human body motions recorded from 30 users performing activities of daily living, with six different classes (i.e., standing, sitting, lying down, normal walking, walking upstairs, and walking downstairs). The users carried waist-mounted smartphones equipped with integrated inertial sensors that measured nine channels (i.e., accelerometers, gyroscopes, and magnetometers, each with three axis) samples at 50 Hz. We evaluate DA among 435 possible combinations of the 30 users.

WISDM. The WISDM (Kwapisz et al., 2010) dataset utilizes three-axis accelerometers embedded in smartphones to recognize six different classes of activities (i.e., walking, jogging, ascending stairs, descending stairs, sitting, and standing) performed by 36 users. We treat each user as a distinct domain and train DA models on 356 possible combinations of the 36 users.

SSC (EEG). The sleep stage classification (SSC) (Goldberger et al., 2000) dataset consists of electroencephalography (EEG) signals representing five different sleep stages (i.e., wakefulness, three non-rapid eye movement stages, and rapid eye movement stage). The dataset also includes univariate heart rate recordings from 20 users, with a length of 3,000 time steps. Each user is treated as a distinct domain, and we train DA models on 190 possible combinations of the 20 users.

uWave. The uWave (Liu et al., 2009) dataset includes gestures performed using a hand-held device equipped with an integrated three-axis accelerometer that allows personalized gestures from a library of five classes. A total of 4,478 samples were recorded from eight users. We interpolate the time-series to 150 time steps and evaluate DA methods on all 28 possible combinations of the eight users. It should be noted that we use a different train/validation (80%/20%) split than previous studies such as Wilson et al. (2020); Liu & Xe (2021).

Finger Movements. The finger movements (Blankertz et al., 2001) dataset consists of EEG recordings from one subject seated in a chair with their fingers placed in a standard typing position on a computer keyboard. The task involves pressing keys with the index

and little fingers, with binary labels indicating *left* and *right* movements. The dataset is relatively small, with only 104 training and testing samples. We train DA models on 378 possible combinations of source and target domain scenarios.

Gestures Mid Air. The gestures mid air (Caputo et al., 2018) dataset is collected using a leap motion device and consists of interface-command gestures from 10 subjects. We train DA models on 45 possible combinations of scenarios.

Epilepsy. The dataset presented by Villar et al. (2016) was generated using a tri-axial accelerometer with three channels samples at 16 Hz, worn on the dominant wrist by six healthy participants performing four different activities (i.e., walking, running, sawing, and seizure mimicking), each at least 10 times. We randomly split the dataset 50%/50% into source and target domains, and further randomly split the data in 50%/50% training and testing sets. Due to the small size of the dataset, samples from all subjects are included in both the source and target domain datasets, but in an unequal manner.

Face Detection. The face detection (Olivetti et al., 2014) dataset addresses the problem to classify between two classes: Whether the participant has been presented with a face image or a scrambled image. The dataset is comprised of MEG recordings (i.e., 144 channels) from 16 participants, containing 9,414 samples.

PenDigits. The PenDigits (Alimoglu & Alpaydin, 1997) dataset (see Figure 1.4a) involves a task of handwritten digit classification (10 classes) based on the 2D coordinates (two channels) of the pen trajectory on a digital screen. A total of 44 writers collected 10,992 samples. Similar to the epilepsy dataset, we randomly divide the datasets into source and target domains.

Online Handwriting. The OnHW database consists of handwritten samples captured using sensor-enhanced pens. The OnHW-chars (Ott et al., 2020b) dataset is a task of recognizing lowercase and uppercase characters (52 classes) and contains 31,275 samples from 119 right-handed writers and 2,270 samples from 9 left-handed writers. The OnHW-symbols and split OnHW-equations datasets (Ott et al., 2022d) consist of single numbers and symbols. The writer’s handedness introduces domain shift, with 27 and 55 right-handed writers, respectively, and 4 left-handed writers for both datasets. We adjust the embeddings between these two domains as they represent the source and target domains. The number of domains in this dataset is lower than the others, but the number of channels (13) and classes (15 and 52) is higher. We evaluate DA in both directions, from left-handed to right-handed writers ($L \rightarrow R$) and from right-handed to left-handed writers ($R \rightarrow L$). For more details, refer to Section 1.2.1.

GNSS-based Interference Detection. The degradation and disruption of GNSS receivers by interference signals have a significant impact on their localization accuracy, which

necessitates their detection and classification (van der Merwe et al., 2023). Furthermore, the successful classification of interference signal waveforms aids in the determination of the signal’s purpose, such as differentiating between interference signals from jammers inside cars or mounted on towers (Raichur et al., 2022). To address this need, Brieger et al. (2022) conducted a study in which wideband snapshots were collected in GNSS bands. These snapshots, taken every second using a sample rate of 62.5 MHz, recorded 20 ms of raw data and captured the real GNSS signal from an outside receiver mixed with interference signals. The GNSS-based dataset was obtained from eight static and dynamic scenarios at varying distances and speeds relative to the receiver and was recorded at the Fraunhofer IIS L.I.N.K. test and application center. As a consequence, the multi-class problem has nine distinct classes: Class 0 corresponds to the non-interference class, while class 1 through 8 represent different types of interference signals to be detected (namely chirp, frequency hopper, modulated, multitone, noise, and pulsed). Alternatively, for the binary classification task, classes 1 to 8 are merged to identify either interference or non-interference signals. To accomplish this, we utilize the following four features out of 88 channels: automatic gain control (AGC), carrier-to-noise density ratio (CN0), azimuth variance, and elevation variance, as suggested by Raichur et al. (2022). To evaluate the dataset, we choose two distinct recording days as domains and adapt from the first to the second. We employ a sliding window, obtained from the *tsai* toolbox¹², with a *stride* of 1, and assess window lengths ranging from 4 to 29. When using a training/test split of 70%/30%, this generates 15,403 training samples and 6,604 testing samples, on average.

Generated Sinusoidal Data. Ott et al. (2022a) generated univariate data using sinusoidal time-series with distinct frequencies for 10 classes and then introduced noise from a continuous uniform distribution $U(a, b)$ with $a = 0.0$ and $b \in \mathcal{B} = \{0.0, 0.1, 0.2, \dots, 1.9\}$ to create the target domain dataset. For the source domain training dataset, we flipped the sign of the time-series and introduced noise $U(a, b/2)$. To assess the effectiveness of DA methods, Ott et al. (2022a) evaluated them using the flipped dataset with noise values $b \in \mathcal{B}$.

Class & Domain Distributions. Methods that utilize a clustering loss on top of embedded features to optimize clustering by pushing data closer to randomly initialized cluster centers may encounter issues when the number of samples varies across different domains or classes (Rezaei et al., 2022). The preponderance of samples in each domain is critical for unsupervised DA. In this regard, we perform a statistical analysis of the dataset. Figure 2.1 and Figure 2.2 provide details on the class distributions and number of samples per participant for all time-series datasets (training and test datasets are combined). For the HHAR dataset (see Figure 2.1a), classes 1 and 4 are underrepresented, while every user contributed equally. Class 2 of the UCI HAR dataset (Figure 2.1b) has 500 fewer samples than classes 4 and 5, with every user contributing equally. In the WISDM dataset (see Figure 2.1c),

¹²Github *tsai*: <https://github.com/timeseriesAI/tsai>

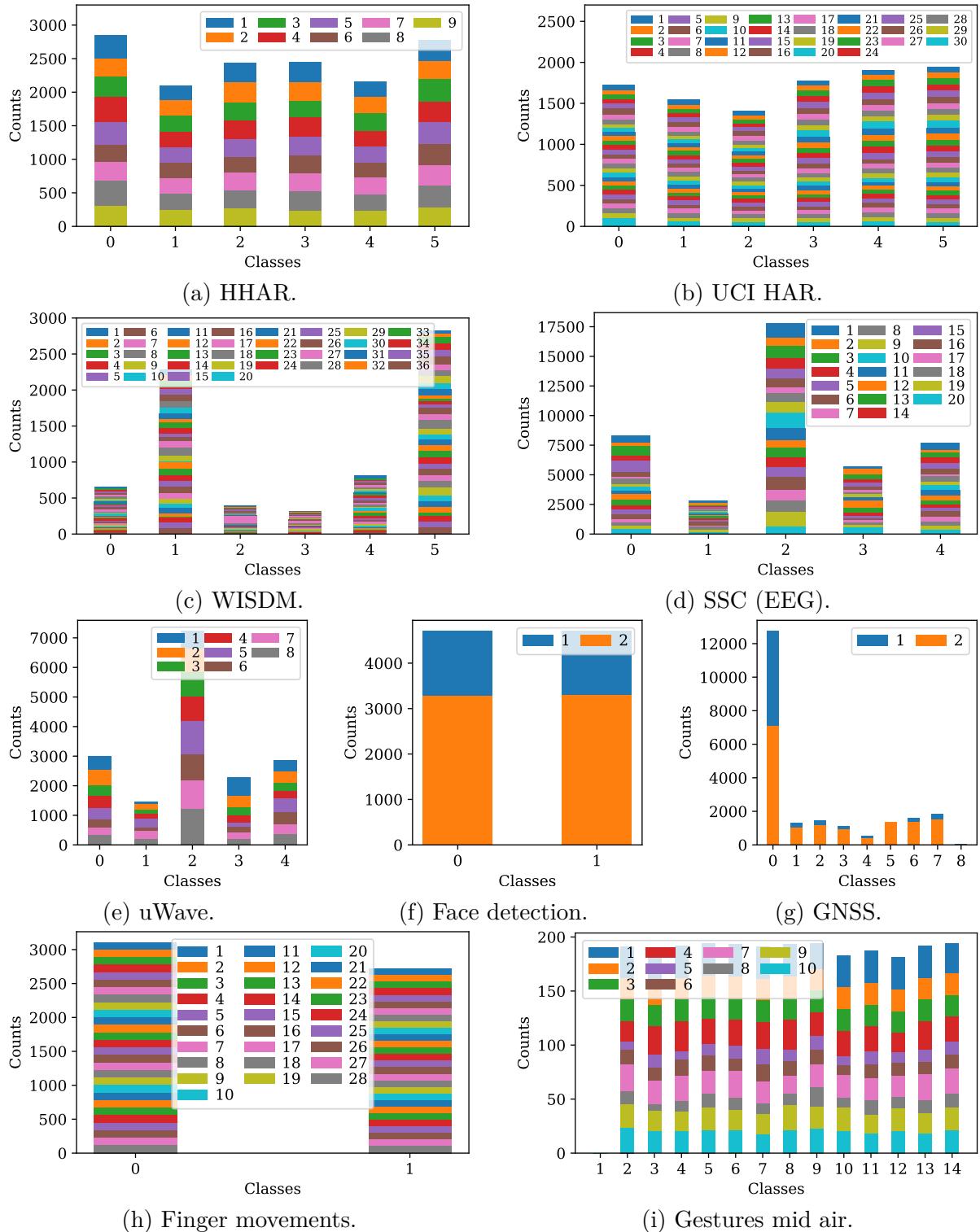


Figure 2.1: Overview of dataset statistics of domain (legend) and class distributions.

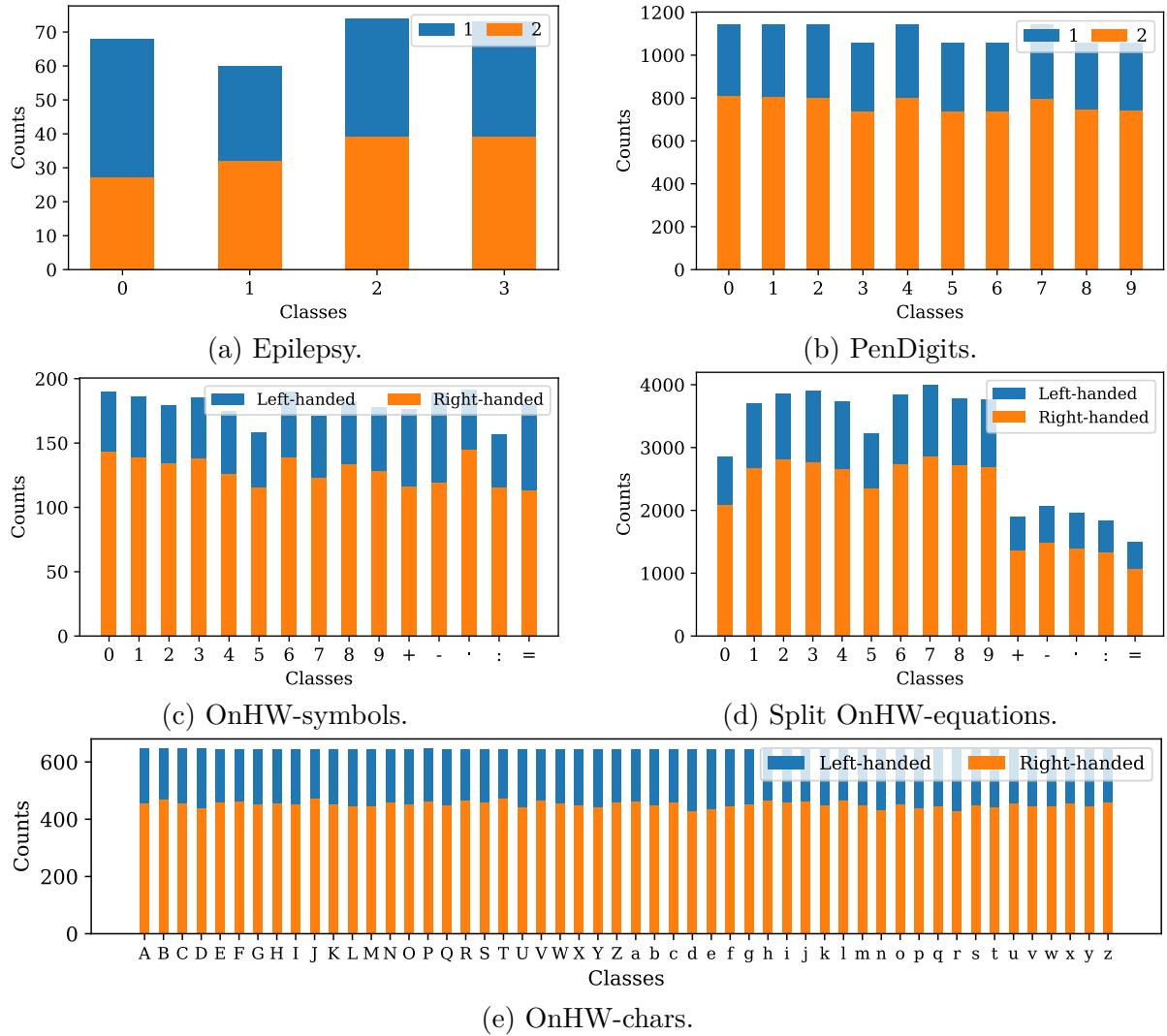


Figure 2.2: Figure 2.1 continued.

classes 1 and 5 preponderate over the classes 0 and 2 to 4 because data collection for specific classes was faster than for other classes. Participants with ID 1 and 2 contributed less data than the remaining participants. The SSC (EEG) dataset (see Figure 2.1d) shows a similar pattern, as class 2 (non-rapid eye movement stage) is predominant in human sleep. Class 2 is also predominant in the uWave dataset (see Figure 2.1e). Since the source and target domain samples are randomly split, the face detection (see Figure 2.1f), epilepsy (see Figure 2.2a), and PenDigits (see Figure 2.2b) datasets are equally distributed between both domains. The task of detecting GNSS-based interference poses a challenge due to the uneven representation of interference classes (see Figure 2.1g for the multi-class detection problem). Class 0 (without interference) contains over 12,000 samples, whereas each class with interference (class 1 to 8) has less than 2,000 samples. Notably, class 8 has very few samples available. In contrast, the binary classification task becomes balanced when the

interference classes are combined with the non-interference classes, and hence, the classification task is unsophisticated. The finger movements dataset (see Figure 2.1h) has an equal contribution from each participant, but this is not the case for the gestures mid air dataset (see Figure 2.1i). Although the classes for OnHW-symbols (see Figure 2.2c) and split OnHW-equations (see Figure 2.2d) are the same, their distributions differ. Regarding OnHW-symbols, each class is distributed approximately equally. Concerning split OnHW-equations, numbers are more likely to appear than operators such as '+', '−', '·', '÷', and '='. Additionally, since the number '0' cannot occur at the beginning of an equation, it appears less frequently than the numbers '1' through '9'. The ratio of right-handed to left-handed writers approximately corresponds to the real-world distribution, meaning that the under-representation of left-handed writers (10.6%) is taken into account¹³. Each writer contributed the same amount of characters to the OnHW-chars dataset, as shown in Figure 2.2e, and as a result, the classes are distributed evenly. In contrast, the classes in the PenDigits handwriting dataset are roughly equally distributed. Given the imbalanced nature of the time-series dataset, we also report macro F1-scores, in addition to the standard accuracy metric, which accounts for the data distribution. The F1-score is defined as

$$\text{F1-score} = \frac{\text{true positive}}{\text{true positive} + \frac{1}{2}(\text{false positive} + \text{false negative})}, \quad (2.3)$$

where true positive is the number of true positives classified by the model, respectively for false positive and false negative.

2.3.2 Architectures & Hyperparameters

Following the approach in AdaTime (Ragab et al., 2023), we assess the performance of three encoder networks in extracting features from samples of the source and target domains. The encoder network is responsible for transforming data from the input space to the feature space. To cover a range of convolutional and spatio-temporal networks (as described in Section 1.3.3), we consider three networks: CNN, ResNet18 (He et al., 2016), and TCN. This selection facilitates a fair comparison of DA methods, although using a more suitable encoder network specific to each dataset could potentially yield better classification results. However, due to computational constraints, we limit our encoder networks to these three options. For the HHAR, UCI HAR, WISDM, SSC, uWave, and generated sinusoidal datasets, we use the hyperparameters proposed by Ragab et al. (2023), while we search for optimal hyperparameters for the OnHW datasets. For the remaining datasets, we select the same hyperparameters as those used for the HAR dataset. We also evaluate normalization of the time-series data as a preprocessing step. When using iterable-style datasets with multi-processing, we set the *drop last* argument to drop the last non-full batch of each dataset, and we search for the optimal *drop last* parameter. In the following, we provide a description of the three encoder networks.

¹³Marietta Papadatou-Pastou, Eleni Ntolka, Judith Schmitz, Maryanne Martin, Marcus R. Munafò, Sebastian Ocklenburg, and Silvia Paracchini. Human Handedness: A Meta-analysis. In *Psychological Bulletin*, volume 146(6), pp. 481-524, June 2020. doi:10.1037/bul0000229.

CNN. The convolutional encoder is composed of three blocks, each comprising a 1D convolutional layer (with 64, 128, and 128 channels, respectively), a batch normalization layer, ReLU activation, and a 1D max pooling layer. Following the three blocks, there is an adaptive average pooling layer (with the parameter *feature length*) and a fully connected layer for the classification task. The first convolutional layer has the hyperparameters *kernel size* and *stride*. The convolutional layers in all blocks are dependent on the hyperparameter *intermediate channels*. The first block includes a dropout with the hyperparameter *dropout rate*. Lastly, the convolutional and batch normalization (Ioffe & Szegedy, 2015) layers in the last block depend on the hyperparameter *final output channels*.

ResNet18. The ResNet architecture, as proposed by He et al. (2016), is composed of four one-dimensional residual blocks and one adaptive average pooling layer (with the parameter *feature length*). Similar to the CNN, we define the parameters for ResNet as *kernel size*, *stride*, *intermediate channels*, and *final output channels*.

TCN. The TCN network comprises a TCN (Bai et al., 2018) layer and two 1D convolutional blocks with ReLU activation.

Loss Parameters. The DA hyperparameters, including the learning rate, source classification loss weighting, domain loss weighting, CORAL weighting, and MMD weighting, were established for the HHAR, UCI HAR, WISDM, and SSC (EEG) datasets as suggested by Ragab et al. (2023). For the remaining datasets, we choose the following hyperparameters for the DA methods¹⁴: MMDA ($lr = 0.001$, $wt_{src} = 6.13$, $wt_{align} = 8.63$, $wt_{MMD} = 2.37$, and $wt_{entropy} = 7.16$), DDC ($lr = 0.005$, $wt_{src} = 6.24$, and $wt_{align} = 6.36$), HoMM ($lr = 0.001$, $wt_{src} = 2.15$, and $wt_{align} = 9.13$), DANN ($lr = 0.01$, $wt_{src} = 9.74$, and $wt_{align} = 5.43$), CDAN ($lr = 0.01$, $wt_{src} = 5.19$, $wt_{align} = 2.91$, and $wt_{entropy} = 1.73$), DIRT-T ($lr = 0.0005$, $wt_{src} = 7.00$, and $wt_{align} = 4.51$, and $wt_{entropy} = 0.79$), DSAN ($lr = 0.0005$, $wt_{src} = 1.76$, and $wt_{align} = 1.59$), CoDATS ($lr = 0.001$, $wt_{src} = 6.21$, and $wt_{align} = 1.72$), AdvSKM ($lr = 0.005$, $wt_{src} = 3.05$, and $wt_{align} = 2.876$), and the same parameters for the remaining methods MSE, CS, PC, KL, JSD, linear MMD, kMMD, Deep CORAL, Jeffreys CORAL, Stein CORAL, linear MMCD, kernelized MMCD, and Sinkhorn (with combinations) ($lr = 0.005$, $wt_{src} = 8.67$, and $wt_{align} = 0.44$). We equally weight Sinkhorn and the second alignment loss function. For kMMD, we search for the hyperparameters *kernel_mul* = [1, 2, 3, 4] and *kernel_num* = [3, 4, 5, 6, 7] of the RBF kernel (Zellinger et al., 2017). For follow-up trainings, we select *kernel_mul* = 2 and *kernel_num* = 5.

¹⁴**Abbreviations.** lr : learning rate, wt_{src} : weighting source classification loss, wt_{align} : weighting domain alignment loss, wt_{MMD} : weighting MMD loss, $wt_{entropy}$: weighting conditional entropy loss.

Table 2.2: Results for the source and target test datasets, which are evaluated on the model trained solely on target samples to show the discrepancy between source and target domains. For all 15 datasets, we report the accuracy and F1-score in % for the CNN, ResNet18, and TCN encoder networks. These metrics are averaged over all domain scenarios and five training runs. We refer to the underlined best-performing encoder network in the main text.

Dataset	CNN				ResNet18				TCN			
	Target		Source		Target		Source		Target		Source	
	Acc.	F1										
HHAR	<u>98.70</u>	98.70	<u>63.88</u>	58.87	98.01	98.05	47.42	42.54	98.02	98.06	60.77	55.83
UCI HAR	<u>99.42</u>	99.48	<u>74.31</u>	71.27	99.77	99.78	69.96	64.99	98.68	98.68	72.44	68.79
WISDM	99.28	98.21	55.23	38.44	99.22	98.09	55.60	32.84	<u>93.83</u>	88.46	<u>56.36</u>	37.25
SSC (EEG)	<u>79.62</u>	70.18	<u>66.69</u>	54.84	62.13	48.34	49.71	36.43	52.45	39.90	45.11	33.52
uWave	<u>99.82</u>	99.83	<u>83.54</u>	82.05	96.43	96.15	43.38	38.12	99.43	99.41	79.65	77.70
Finger movements	55.91	51.08	51.23	42.55	50.66	45.47	50.51	46.00	<u>57.00</u>	55.16	<u>51.48</u>	46.43
Gestures mid air	67.14	65.11	42.80	37.46	64.23	61.82	38.48	33.97	<u>69.84</u>	67.46	<u>45.89</u>	41.58
Epilepsy	80.31	75.40	77.50	72.87	<u>95.31</u>	95.33	<u>97.81</u>	97.87	74.38	68.06	74.69	73.30
Face detection	52.59	40.59	56.46	48.86	53.52	50.94	51.75	45.88	<u>69.92</u>	69.92	<u>64.10</u>	64.09
PenDigits	<u>99.69</u>	99.68	<u>96.07</u>	95.99	84.41	84.28	78.92	78.78	99.54	99.53	96.03	96.01
OnHW-symbols	<u>77.19</u>	78.07	20.00	14.87	76.25	76.59	13.75	9.74	19.69	15.12	5.00	3.54
Split OnHW-equations	<u>89.66</u>	89.02	<u>36.83</u>	38.29	82.34	82.30	21.92	21.60	56.94	54.48	29.49	29.44
OnHW-chars (combined)	<u>73.52</u>	71.76	<u>37.44</u>	36.83	49.93	48.02	16.10	15.05	1.29	0.86	1.92	0.07
GNSS (binary)	<u>97.16</u>	95.61	<u>73.69</u>	54.81	95.48	93.01	66.26	60.57	96.86	95.14	69.22	54.61
GNSS (multi-class)	<u>73.27</u>	10.85										
Sinusoidal data ($b = 0.0$)	23.99	15.24	40.12	29.18	58.69	50.45	10.18	3.45	<u>100.0</u>	100.0	<u>0.00</u>	0.00
Sinusoidal data ($b = 0.5$)	46.27	39.35	43.55	32.50	45.56	36.72	10.91	5.12	<u>98.35</u>	98.33	<u>8.65</u>	7.09
Sinusoidal data ($b = 1.0$)	34.74	26.95	37.28	29.55	21.01	9.65	8.67	2.33	<u>84.92</u>	84.24	<u>0.87</u>	1.12
Sinusoidal data ($b = 1.5$)	25.79	17.57	30.97	25.37	11.11	2.79	8.41	3.59	<u>65.04</u>	62.32	<u>0.38</u>	0.44
Sinusoidal data ($b = 1.9$)	28.75	28.95	22.69	24.17	10.30	2.30	8.87	3.69	<u>59.05</u>	57.58	<u>0.50</u>	0.59

2.4 Evaluation Results

Hardware & Training Setup. For all experiments, we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We utilize the vanilla Adam optimizer with a weight decay of 10^{-4} , $\beta_1 = 0.5$, and $\beta_2 = 0.99$. Each dataset and method is trained for 40 epochs and the results are averaged over five trainings. For the SSC (EEG) dataset, we use a batch size of 128, whereas for the remaining datasets, a batch size of 32 is used. In the following work, we examine the domain gap between the source and target domains and provide a detailed analysis of the evaluation results for the DA methods and encoder networks. Furthermore, we propose hyperparameter searches for the Sinkhorn and OnHW datasets, and assess the associated risks.

Discrepancy Between Source and Target Domains. We begin by assessing the domain shift between the source and target domains, which involves training and testing the models on either the target domain dataset or training the model on the source domain

Table 2.3: DA results in % (mean and standard deviation over all scenarios and five runs) for the HHAR (Stisen et al., 2015) dataset utilizing AdaTime. **Bold** are best results. Table 2.4 to Table 2.12 continues.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	62.24 \pm 16.7	57.79 \pm 18.0	48.40 \pm 16.4	43.37 \pm 18.1	52.08 \pm 14.1	46.31 \pm 16.2
CS	63.07 \pm 18.7	59.09 \pm 21.0	53.68 \pm 19.6	50.98 \pm 22.1	56.09 \pm 17.8	51.36 \pm 20.8
PC	64.74 \pm 17.3	60.68 \pm 19.1	54.97 \pm 19.6	51.59 \pm 22.1	61.43 \pm 16.4	57.20 \pm 19.1
KL	60.10 \pm 16.9	54.76 \pm 18.6	50.83 \pm 18.7	46.08 \pm 21.7	19.36 \pm 6.0	8.39 \pm 5.8
JSD	64.75 \pm 17.4	60.62 \pm 20.0	54.55 \pm 20.4	50.74 \pm 23.4	17.79 \pm 3.6	7.33 \pm 3.1
Linear MMD	66.83 \pm 17.5	63.13 \pm 19.7	57.50 \pm 20.3	54.95 \pm 22.8	68.33 \pm 18.3	65.01 \pm 21.3
kMMD	63.86 \pm 17.1	59.64 \pm 18.9	54.50 \pm 19.8	51.25 \pm 22.3	61.48 \pm 17.4	57.48 \pm 20.6
Deep CORAL	73.51 \pm 17.2	71.57 \pm 18.9	61.20 \pm 19.2	59.75 \pm 21.1	72.17 \pm 16.8	70.23 \pm 18.9
Jeffreys CORAL	64.49 \pm 17.5	60.18 \pm 19.4	54.55 \pm 19.7	61.11 \pm 22.3	62.13 \pm 17.7	58.15 \pm 21.0
Stein CORAL	64.10 \pm 16.9	59.78 \pm 18.9	54.93 \pm 19.7	51.57 \pm 22.2	62.11 \pm 16.6	58.18 \pm 19.4
Linear MMCD	66.60 \pm 17.5	63.03 \pm 19.7	57.32 \pm 20.2	54.87 \pm 22.7	68.04 \pm 18.2	64.78 \pm 21.1
Kernelized MMCD	64.37 \pm 17.1	60.11 \pm 19.0	55.26 \pm 19.7	52.06 \pm 22.3	62.03 \pm 17.2	57.97 \pm 20.4
MMDA	70.34 \pm 14.0	66.08 \pm 15.0	57.73 \pm 20.8	55.05 \pm 23.5	67.26 \pm 17.6	64.51 \pm 20.1
DDC	63.14 \pm 16.6	59.32 \pm 19.1	53.74 \pm 17.7	49.80 \pm 20.3	59.70 \pm 16.0	55.69 \pm 18.9
Linear DAN	72.14 \pm 18.5	69.81 \pm 20.6	59.19 \pm 19.9	56.96 \pm 22.3	68.40 \pm 16.9	65.86 \pm 19.5
Squared DAN	72.09 \pm 18.4	69.84 \pm 20.3	59.29 \pm 20.2	57.09 \pm 22.6	68.48 \pm 17.7	66.12 \pm 20.1
HoMM _{p=3}	72.64 \pm 17.3	70.15 \pm 19.4	61.59 \pm 19.8	59.70 \pm 22.4	70.84 \pm 16.8	68.48 \pm 19.3
DANN	76.91 \pm 18.9	75.86 \pm 19.6	63.21 \pm 20.3	62.18 \pm 21.5	73.16 \pm 18.3	71.86 \pm 19.4
CDAN	79.88 \pm 18.2	79.44 \pm 18.5	66.49 \pm 20.4	66.11 \pm 20.9	76.78 \pm 16.7	75.10 \pm 18.2
DIRT-T	80.11 \pm 18.3	79.45 \pm 18.8	67.89 \pm 22.0	65.92 \pm 23.5	76.78 \pm 17.5	74.77 \pm 19.4
DSAN	77.96 \pm 19.6	77.28 \pm 20.4	67.22 \pm 22.2	66.60 \pm 23.1	74.64 \pm 18.7	73.26 \pm 20.3
CoDATS	76.30 \pm 19.6	75.62 \pm 20.4	63.79 \pm 19.1	62.73 \pm 20.0	72.48 \pm 18.1	71.10 \pm 19.5
AdvSKM	66.36 \pm 15.3	62.23 \pm 17.0	53.61 \pm 18.6	50.28 \pm 20.7	62.32 \pm 15.8	58.07 \pm 18.3
Sinkhorn	75.15 \pm 19.5	73.97 \pm 20.7	67.98 \pm 20.9	67.18 \pm 22.5	70.04 \pm 19.1	68.11 \pm 21.4

dataset and evaluating it on the target domain dataset. The results are presented in Table 2.2. Across all datasets, we observe a domain gap between the source and target data, with the HHAR dataset exhibiting a domain shift of 34.82%p (percent points), the UCI HAR dataset of 25.11%p, the WISDM dataset of 37.47%p, the SSC dataset of 12.93%p, the uWave dataset of 16.28%p, and the gestures mid air dataset of 23.95%p. Hence, regardless of the dataset, a domain shift exists. Moreover, the domain shift is even more pronounced for the OnHW datasets (57.19%p, 52.83%p, and 36.08%p), as the sensor data differs significantly between right-handed and left-handed writers, as reported in Klaß et al. (2022). For the finger movements, epilepsy, face detection, and PenDigits datasets, which are randomly shuffled, the domain gap is small. However, for GNSS interference detection (with a window length of 14), a domain shift is observed only in the binary case, with a shift of 23.47%p. In the multi-class problem, the models tend to predict the class “non-interference” regardless of the domain. In the sinusoidal dataset, the classification accuracy significantly decreases with an increased level of added noise, as demonstrated in Ott et al. (2022a, 2023c). The accuracy drops from 100% for $b = 0.0$ to 59.05% for $b = 1.9$ on the target domain data. Since the model is unable to extract any meaningful features of the source domain that cannot be correctly classified using the model trained on the target domain, there is a substantial shift between the two domains. The gap is reduced when using the CNN encoder but at the expense of lower target accuracy.

Table 2.4: UCI HAR (Anguita et al., 2013) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	72.41 \pm 18.0	69.14 \pm 19.9	68.04 \pm 16.4	63.93 \pm 17.7	69.73 \pm 16.0	65.92 \pm 17.3
CS	77.62 \pm 18.0	75.43 \pm 19.9	73.88 \pm 16.3	70.49 \pm 17.7	68.99 \pm 16.5	65.34 \pm 17.8
PC	80.52 \pm 15.9	78.79 \pm 17.1	77.51 \pm 15.5	74.53 \pm 16.8	78.26 \pm 14.6	76.08 \pm 15.6
KL	69.17 \pm 18.9	65.30 \pm 21.1	72.04 \pm 16.4	67.42 \pm 18.0	16.64 \pm 4.8	5.36 \pm 3.2
JSD	79.70 \pm 16.3	77.95 \pm 17.5	72.97 \pm 14.7	67.67 \pm 16.5	15.85 \pm 3.8	4.74 \pm 1.9
Linear MMD	82.48 \pm 14.7	81.22 \pm 15.7	80.97 \pm 13.4	78.44 \pm 14.6	84.62 \pm 12.6	83.35 \pm 13.6
kMMD	80.95 \pm 15.7	79.26 \pm 17.0	77.88 \pm 15.2	74.94 \pm 16.6	78.46 \pm 14.8	76.28 \pm 15.8
Deep CORAL	80.93 \pm 14.5	79.20 \pm 15.5	79.26 \pm 13.2	76.60 \pm 14.1	78.95 \pm 12.7	76.77 \pm 13.4
Jeffreys CORAL	80.99 \pm 15.6	79.32 \pm 16.9	77.90 \pm 15.3	74.97 \pm 16.6	78.33 \pm 14.7	76.10 \pm 15.9
Stein CORAL	80.99 \pm 15.6	79.32 \pm 16.8	77.92 \pm 15.1	75.00 \pm 16.5	78.52 \pm 14.7	76.37 \pm 15.7
Linear MMCD	82.45 \pm 14.7	81.20 \pm 15.6	81.11 \pm 13.5	78.67 \pm 14.7	84.62 \pm 12.8	82.33 \pm 13.9
Kernelized MMCD	80.93 \pm 15.7	79.20 \pm 16.9	79.26 \pm 14.4	76.60 \pm 15.6	78.95 \pm 14.2	76.77 \pm 15.3
MMDA	89.92 \pm 11.4	89.86 \pm 11.5	79.41 \pm 13.7	77.04 \pm 14.8	77.75 \pm 12.9	75.35 \pm 13.5
DDC	80.96 \pm 14.5	79.29 \pm 15.4	77.88 \pm 14.0	74.96 \pm 14.9	78.57 \pm 13.2	76.40 \pm 13.9
Linear DAN	86.17 \pm 12.3	85.50 \pm 12.9	81.91 \pm 13.1	80.01 \pm 14.1	81.73 \pm 13.1	80.15 \pm 14.0
Squared DAN	86.31 \pm 12.5	85.69 \pm 13.0	82.22 \pm 13.2	80.41 \pm 14.2	82.42 \pm 13.0	81.03 \pm 13.8
HoMM _{p=3}	88.41 \pm 11.9	88.39 \pm 11.8	84.72 \pm 12.0	83.43 \pm 12.8	87.90 \pm 10.7	87.64 \pm 10.8
DANN	86.97 \pm 10.4	86.33 \pm 10.6	80.20 \pm 12.6	78.03 \pm 13.4	84.15 \pm 10.8	83.03 \pm 11.3
CDAN	89.15 \pm 10.5	88.44 \pm 11.0	85.16 \pm 12.2	83.20 \pm 13.2	83.36 \pm 10.9	81.45 \pm 11.5
DIRT-T	92.02 \pm 9.6	91.97 \pm 10.0	80.58 \pm 14.5	76.04 \pm 18.1	85.78 \pm 13.1	83.69 \pm 15.9
DSAN	89.25 \pm 11.2	88.65 \pm 11.7	88.17 \pm 12.1	87.65 \pm 12.5	87.35 \pm 11.2	86.51 \pm 11.6
CoDATS	87.38 \pm 11.7	86.78 \pm 12.0	83.11 \pm 12.6	81.72 \pm 13.3	85.56 \pm 11.5	84.75 \pm 11.9
AdvSKM	80.77 \pm 14.5	78.95 \pm 15.4	77.98 \pm 14.6	75.23 \pm 15.5	78.71 \pm 12.9	76.55 \pm 13.6
Sinkhorn	87.17 \pm 12.4	86.55 \pm 12.9	81.62 \pm 12.7	78.96 \pm 14.0	80.88 \pm 12.9	79.05 \pm 13.5

Table 2.5: WISDM (Kwapisz et al., 2010) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	49.91 \pm 19.8	33.28 \pm 18.4	48.92 \pm 16.8	27.87 \pm 14.4	50.19 \pm 15.1	27.08 \pm 12.9
CS	50.69 \pm 20.2	34.60 \pm 18.2	50.84 \pm 19.9	30.87 \pm 16.5	39.26 \pm 15.0	25.93 \pm 13.6
PC	58.21 \pm 21.2	40.51 \pm 20.6	57.60 \pm 18.8	35.47 \pm 17.4	53.79 \pm 20.3	37.50 \pm 19.5
KL	45.27 \pm 17.6	30.57 \pm 17.2	54.72 \pm 18.7	32.75 \pm 17.1	29.34 \pm 12.3	9.66 \pm 4.8
JSD	58.46 \pm 20.0	41.23 \pm 20.6	54.71 \pm 20.4	35.28 \pm 19.2	29.03 \pm 11.9	8.75 \pm 4.2
Linear MMD	59.68 \pm 19.9	41.04 \pm 20.1	58.28 \pm 18.7	36.27 \pm 17.8	60.37 \pm 19.7	38.93 \pm 18.9
kMMD	59.15 \pm 20.6	41.28 \pm 20.6	57.70 \pm 18.6	35.43 \pm 17.2	57.40 \pm 18.9	38.13 \pm 18.7
Deep CORAL	60.21 \pm 19.7	41.91 \pm 20.1	58.84 \pm 17.8	36.69 \pm 16.9	59.15 \pm 18.4	39.39 \pm 18.4
Jeffreys CORAL	59.18 \pm 20.6	41.30 \pm 20.6	57.68 \pm 18.6	35.43 \pm 17.2	57.34 \pm 18.9	38.09 \pm 18.7
Stein CORAL	59.15 \pm 20.6	41.29 \pm 20.6	57.70 \pm 18.7	35.44 \pm 17.2	57.37 \pm 18.9	38.08 \pm 18.7
Linear MMCD	59.73 \pm 19.9	41.10 \pm 20.1	58.58 \pm 18.8	36.53 \pm 17.9	60.41 \pm 19.7	38.95 \pm 18.9
Kernelized MMCD	59.25 \pm 20.6	41.34 \pm 20.6	58.13 \pm 18.6	35.85 \pm 17.5	57.87 \pm 18.8	38.40 \pm 18.6
MMDA	62.46 \pm 20.1	43.43 \pm 20.1	55.27 \pm 18.3	31.28 \pm 14.8	51.95 \pm 15.4	29.28 \pm 14.4
DDC	59.68 \pm 19.4	41.05 \pm 19.8	57.77 \pm 17.9	34.34 \pm 15.9	59.08 \pm 18.7	38.74 \pm 18.3
Linear DAN	63.07 \pm 20.8	43.62 \pm 21.5	60.75 \pm 18.9	38.22 \pm 18.5	59.81 \pm 18.9	39.65 \pm 19.4
Squared DAN	63.15 \pm 20.7	43.66 \pm 21.5	61.01 \pm 18.9	38.67 \pm 18.8	60.58 \pm 19.0	40.43 \pm 19.7
HoMM _{p=3}	42.41 \pm 21.8	42.40 \pm 21.8	59.32 \pm 17.9	35.51 \pm 17.3	59.89 \pm 18.9	40.84 \pm 19.6
DANN	63.37 \pm 20.2	45.16 \pm 21.5	58.26 \pm 18.5	39.73 \pm 19.1	58.15 \pm 17.7	39.89 \pm 18.5
CDAN	61.02 \pm 19.7	40.95 \pm 19.6	56.85 \pm 17.7	34.31 \pm 15.5	59.74 \pm 18.5	39.10 \pm 18.1
DIRT-T	61.32 \pm 17.6	34.47 \pm 14.1	55.76 \pm 15.5	27.46 \pm 10.8	58.18 \pm 17.7	31.25 \pm 13.9
DSAN	55.40 \pm 18.8	30.80 \pm 13.9	59.77 \pm 17.9	34.62 \pm 16.1	54.98 \pm 18.8	34.77 \pm 16.3
CoDATS	62.77 \pm 19.8	43.74 \pm 20.4	59.34 \pm 18.8	39.16 \pm 19.0	38.68 \pm 18.6	36.68 \pm 18.6
AdvSKM	59.36 \pm 19.7	40.79 \pm 20.0	57.26 \pm 17.9	33.91 \pm 15.6	58.11 \pm 18.9	38.33 \pm 18.3
Sinkhorn	59.40 \pm 22.4	41.15 \pm 22.4	59.18 \pm 20.3	40.28 \pm 20.8	55.66 \pm 18.2	35.20 \pm 17.4

Evaluation of DA & Encoder Networks on the Human Activity Datasets. Table 2.3 to Table 2.7 present the DA results for the HHAR, UCI HAR, WISDM, SSC (EEG),

Table 2.6: SSC (EEG) (Goldberger et al., 2000) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	69.12 \pm 11.1	58.02 \pm 12.1	51.63 \pm 9.1	38.55 \pm 8.8	45.87 \pm 4.8	33.59 \pm 4.7
CS	71.51 \pm 9.8	57.79 \pm 11.3	53.13 \pm 8.1	36.18 \pm 9.2	46.00 \pm 4.9	33.89 \pm 4.8
PC	69.80 \pm 11.1	59.03 \pm 11.6	51.86 \pm 9.1	38.80 \pm 8.7	45.89 \pm 5.0	34.15 \pm 4.9
KL	69.20 \pm 11.0	58.38 \pm 11.7	50.03 \pm 9.5	37.36 \pm 8.7	26.56 \pm 14.3	8.21 \pm 3.7
JSD	69.99 \pm 10.5	59.30 \pm 11.0	48.80 \pm 10.2	35.65 \pm 9.2	26.47 \pm 14.1	8.21 \pm 3.7
Linear MMD	70.18 \pm 10.4	59.39 \pm 11.1	51.74 \pm 9.2	38.73 \pm 8.8	46.16 \pm 5.2	33.78 \pm 5.3
kMMD	69.80 \pm 10.8	59.11 \pm 11.4	51.91 \pm 9.2	38.96 \pm 8.8	45.98 \pm 5.0	34.00 \pm 4.9
Deep CORAL	72.44 \pm 8.4	61.45 \pm 8.6	52.67 \pm 8.3	39.21 \pm 8.1	46.79 \pm 4.4	33.54 \pm 4.6
Jeffreys CORAL	69.69 \pm 10.8	58.93 \pm 11.6	51.77 \pm 9.3	38.81 \pm 8.8	45.93 \pm 4.9	34.01 \pm 4.8
Stein CORAL	69.95 \pm 11.0	59.24 \pm 11.5	51.87 \pm 9.0	38.90 \pm 8.7	45.96 \pm 4.9	34.08 \pm 4.8
Linear MMCD	70.19 \pm 10.3	59.35 \pm 11.0	51.82 \pm 9.4	38.77 \pm 8.9	45.85 \pm 5.3	33.55 \pm 5.3
Kernelized MMCD	69.83 \pm 11.3	59.04 \pm 11.9	51.74 \pm 9.4	38.80 \pm 8.9	45.92 \pm 5.0	33.88 \pm 5.1
MMDA	64.84 \pm 8.5	64.84 \pm 8.5	52.15 \pm 7.8	36.35 \pm 8.4	44.48 \pm 5.1	28.31 \pm 5.2
DDC	72.44 \pm 8.4	61.44 \pm 8.6	51.32 \pm 8.0	37.98 \pm 8.5	46.77 \pm 4.5	33.45 \pm 4.6
Linear DAN	70.25 \pm 10.7	59.70 \pm 11.2	51.78 \pm 9.3	38.82 \pm 8.9	45.98 \pm 4.9	34.01 \pm 4.8
Squared DAN	70.50 \pm 10.5	59.79 \pm 11.0	52.02 \pm 9.0	39.02 \pm 8.7	45.96 \pm 4.9	34.10 \pm 4.8
HoMM _{p=3}	72.84 \pm 8.5	61.74 \pm 8.7	51.16 \pm 8.1	37.00 \pm 8.2	40.42 \pm 6.3	19.14 \pm 4.7
DANN	73.09 \pm 8.5	62.00 \pm 8.5	52.54 \pm 8.1	38.54 \pm 7.8	46.30 \pm 4.5	33.00 \pm 4.6
CDAN	71.70 \pm 10.5	58.10 \pm 11.7	49.90 \pm 8.2	32.28 \pm 9.2	47.88 \pm 5.0	33.47 \pm 5.5
DIRT-T	73.48 \pm 10.3	61.23 \pm 11.1	52.37 \pm 7.7	37.38 \pm 8.0	47.85 \pm 5.1	33.30 \pm 6.1
DSAN	71.00 \pm 10.4	58.15 \pm 10.5	51.53 \pm 8.3	36.27 \pm 9.3	47.43 \pm 4.7	34.71 \pm 5.2
CoDATS	70.03 \pm 8.5	58.34 \pm 8.8	50.67 \pm 7.4	35.63 \pm 7.3	42.39 \pm 5.7	26.78 \pm 5.4
AdvSKM	72.56 \pm 8.8	61.37 \pm 9.1	50.97 \pm 7.5	35.32 \pm 7.8	46.62 \pm 4.5	33.35 \pm 4.7
Sinkhorn	71.37 \pm 10.1	59.97 \pm 10.8	51.31 \pm 8.9	37.15 \pm 8.7	44.87 \pm 6.2	28.48 \pm 6.3

Table 2.7: uWave (Liu et al., 2009) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	83.76 \pm 9.6	82.43 \pm 10.3	45.88 \pm 12.0	41.66 \pm 12.3	71.69 \pm 13.0	68.82 \pm 14.3
CS	77.88 \pm 12.6	76.09 \pm 13.6	48.51 \pm 11.5	44.81 \pm 12.0	69.08 \pm 14.9	66.29 \pm 15.5
PC	85.62 \pm 8.9	84.62 \pm 9.6	49.99 \pm 12.0	45.99 \pm 12.9	73.33 \pm 14.4	70.22 \pm 15.4
KL	81.35 \pm 11.5	79.89 \pm 12.4	46.41 \pm 11.4	40.81 \pm 12.0	12.69 \pm 2.6	2.81 \pm 0.5
JSD	84.10 \pm 11.2	82.72 \pm 12.1	49.33 \pm 10.6	43.11 \pm 11.5	12.10 \pm 3.0	2.70 \pm 0.6
Linear MMD	86.53 \pm 10.1	85.58 \pm 10.7	49.79 \pm 11.7	45.79 \pm 12.4	83.40 \pm 9.9	81.69 \pm 10.9
kMMD	86.57 \pm 9.3	85.53 \pm 10.1	49.76 \pm 12.0	45.78 \pm 12.7	79.22 \pm 13.2	77.28 \pm 14.0
Deep CORAL	86.51 \pm 8.1	85.44 \pm 8.9	50.11 \pm 10.8	46.16 \pm 11.6	80.15 \pm 12.5	78.30 \pm 13.5
Jeffreys CORAL	85.98 \pm 9.9	84.88 \pm 10.7	49.75 \pm 11.7	45.61 \pm 12.4	79.44 \pm 13.3	77.50 \pm 14.2
Stein CORAL	86.41 \pm 9.3	85.41 \pm 9.9	49.81 \pm 12.1	45.71 \pm 12.8	79.11 \pm 13.4	77.15 \pm 14.2
Linear MMCD	86.39 \pm 9.7	85.26 \pm 10.6	49.75 \pm 12.0	45.73 \pm 12.6	83.68 \pm 10.5	81.84 \pm 11.8
Kernelized MMCD	86.51 \pm 9.1	85.44 \pm 10.1	50.11 \pm 11.8	46.16 \pm 12.5	80.15 \pm 13.0	78.30 \pm 14.0
MMDA	92.26 \pm 6.2	91.76 \pm 6.9	49.90 \pm 10.6	46.24 \pm 11.0	84.64 \pm 11.3	82.61 \pm 13.0
DDC	86.32 \pm 8.1	85.28 \pm 8.8	49.96 \pm 10.4	45.96 \pm 11.2	79.43 \pm 12.4	77.42 \pm 13.3
Linear DAN	87.77 \pm 8.7	87.03 \pm 9.4	50.58 \pm 11.0	46.59 \pm 11.7	83.46 \pm 11.9	82.02 \pm 12.7
Squared DAN	89.53 \pm 8.0	88.83 \pm 8.7	51.03 \pm 11.1	47.03 \pm 11.9	86.84 \pm 11.3	85.98 \pm 12.0
HoMM _{p=3}	92.00 \pm 6.1	91.27 \pm 7.1	51.28 \pm 10.2	47.62 \pm 11.1	90.36 \pm 9.8	89.90 \pm 10.2
DANN	85.13 \pm 8.2	84.13 \pm 8.7	52.01 \pm 10.2	48.26 \pm 11.0	84.58 \pm 11.0	83.37 \pm 11.9
CDAN	96.43 \pm 5.5	95.97 \pm 6.9	57.95 \pm 9.2	53.18 \pm 10.7	81.85 \pm 12.2	79.01 \pm 13.7
DIRT-T	96.43 \pm 5.5	95.97 \pm 6.9	55.01 \pm 12.9	49.97 \pm 13.6	80.15 \pm 12.5	78.30 \pm 13.5
DSAN	13.11 \pm 2.7	5.45 \pm 1.6	59.25 \pm 9.4	56.85 \pm 10.4	91.92 \pm 10.9	91.55 \pm 11.6
CoDATS	88.85 \pm 8.0	88.32 \pm 8.8	55.48 \pm 8.3	52.19 \pm 9.6	89.31 \pm 8.1	88.46 \pm 8.9
AdvSKM	85.54 \pm 9.1	84.61 \pm 9.7	50.07 \pm 11.6	46.18 \pm 12.0	79.74 \pm 12.5	77.68 \pm 13.5
Sinkhorn	92.89 \pm 7.3	91.41 \pm 8.2	58.01 \pm 10.0	54.21 \pm 11.3	84.08 \pm 13.2	83.01 \pm 14.0

and uWave datasets for the CNN, TCN, and ResNet18 encoder networks. For the HHAR dataset, DIRT-T using the CNN encoder outperforms other methods, achieving an accu-

Table 2.8: Finger (Blankertz et al., 2001) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	50.18 \pm 4.5	45.62 \pm 5.1	49.58 \pm 4.3	43.72 \pm 4.9	53.66 \pm 4.0	46.03 \pm 7.1
CS	53.27 \pm 1.6	40.34 \pm 3.5	53.47 \pm 2.0	39.03 \pm 4.5	53.26 \pm 4.2	49.53 \pm 5.6
PC	50.50 \pm 4.4	45.71 \pm 5.2	49.69 \pm 4.4	43.14 \pm 5.5	52.78 \pm 4.6	48.21 \pm 5.8
KL	50.18 \pm 4.6	45.53 \pm 5.2	51.26 \pm 4.2	41.59 \pm 5.4	50.90 \pm 5.0	36.00 \pm 6.8
JSD	50.18 \pm 4.5	45.68 \pm 5.2	51.23 \pm 4.0	41.48 \pm 5.9	50.80 \pm 5.0	35.69 \pm 6.4
Linear MMD	50.56 \pm 4.3	45.96 \pm 5.0	49.51 \pm 4.3	43.56 \pm 5.0	54.48 \pm 2.7	39.47 \pm 5.6
kMMD	50.36 \pm 4.3	45.97 \pm 5.1	49.67 \pm 4.4	43.28 \pm 5.5	52.63 \pm 4.6	48.05 \pm 5.8
Deep CORAL	50.35 \pm 4.4	45.94 \pm 5.2	49.68 \pm 4.4	43.30 \pm 5.5	52.77 \pm 4.8	48.50 \pm 5.9
Jeffreys CORAL	50.36 \pm 4.4	45.95 \pm 5.2	49.64 \pm 4.4	43.33 \pm 5.4	52.80 \pm 4.5	48.11 \pm 5.8
Stein CORAL	50.44 \pm 4.4	46.01 \pm 5.2	49.68 \pm 4.3	43.43 \pm 5.4	52.85 \pm 4.6	48.31 \pm 5.8
Linear MMCD	50.58 \pm 4.4	45.94 \pm 5.2	49.49 \pm 4.3	43.53 \pm 4.9	54.49 \pm 2.7	39.33 \pm 5.5
Kernelized MMCD	50.35 \pm 4.4	45.94 \pm 5.2	49.68 \pm 4.4	43.30 \pm 5.5	52.77 \pm 4.8	48.50 \pm 5.9
MMDA	50.69 \pm 4.6	46.04 \pm 5.2	51.36 \pm 3.8	44.88 \pm 5.0	51.60 \pm 4.4	45.41 \pm 6.2
DDC	50.29 \pm 4.4	46.03 \pm 5.1	49.58 \pm 4.4	43.35 \pm 5.3	52.89 \pm 4.6	48.71 \pm 5.9
Linear DAN	50.64 \pm 4.4	45.81 \pm 5.0	49.66 \pm 4.4	43.05 \pm 5.4	52.63 \pm 4.6	48.24 \pm 5.8
Squared DAN	50.54 \pm 4.3	46.01 \pm 5.0	49.54 \pm 4.4	43.31 \pm 5.4	52.52 \pm 4.8	48.15 \pm 5.8
HoMM _{p=3}	49.82 \pm 4.9	43.31 \pm 6.4	49.99 \pm 4.6	44.72 \pm 4.9	53.16 \pm 4.0	49.86 \pm 4.8
DANN	48.74 \pm 4.3	46.33 \pm 5.1	48.17 \pm 4.1	36.86 \pm 5.5	52.75 \pm 4.3	48.92 \pm 6.0
CDAN	51.53 \pm 4.4	46.18 \pm 5.3	50.01 \pm 4.1	42.56 \pm 5.1	53.34 \pm 3.5	50.31 \pm 5.2
DIRT-T	51.02 \pm 5.0	42.49 \pm 6.1	49.34 \pm 4.2	47.35 \pm 4.1	51.45 \pm 4.3	45.53 \pm 6.2
DSAN	52.19 \pm 3.3	39.81 \pm 5.5	49.44 \pm 4.4	45.53 \pm 4.9	51.32 \pm 4.9	49.80 \pm 4.8
CoDATS	51.74 \pm 4.1	44.71 \pm 5.5	50.43 \pm 4.1	43.84 \pm 5.5	52.98 \pm 4.3	49.00 \pm 5.4
AdvSKM	50.47 \pm 4.6	46.03 \pm 5.7	47.92 \pm 4.0	44.70 \pm 4.6	52.22 \pm 4.6	48.37 \pm 5.7
Sinkhorn	50.35 \pm 4.9	46.11 \pm 4.9	48.33 \pm 4.2	42.32 \pm 6.1	55.07 \pm 1.1	35.88 \pm 2.2

Table 2.9: Gestures mid air (Caputo et al., 2018) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	42.56 \pm 14.5	37.83 \pm 14.9	37.32 \pm 14.3	32.65 \pm 14.7	43.04 \pm 15.0	38.05 \pm 15.8
CS	30.52 \pm 12.4	23.83 \pm 13.1	35.48 \pm 14.5	30.39 \pm 15.0	20.26 \pm 13.1	13.08 \pm 13.6
PC	41.96 \pm 14.2	37.12 \pm 14.8	37.21 \pm 14.9	32.43 \pm 15.2	43.49 \pm 14.7	38.18 \pm 15.6
KL	42.23 \pm 14.3	37.40 \pm 15.0	35.60 \pm 14.2	30.46 \pm 14.1	26.26 \pm 19.5	20.44 \pm 19.7
JSD	41.50 \pm 14.8	36.56 \pm 15.4	34.97 \pm 13.0	29.57 \pm 13.0	8.78 \pm 8.3	2.58 \pm 7.9
Linear MMD	42.43 \pm 14.3	37.47 \pm 14.7	37.03 \pm 14.9	32.30 \pm 15.1	40.42 \pm 14.4	35.25 \pm 14.8
kMMD	41.81 \pm 14.7	37.10 \pm 15.1	37.73 \pm 14.3	32.72 \pm 14.6	42.78 \pm 15.0	37.80 \pm 15.7
Deep CORAL	41.30 \pm 14.5	36.36 \pm 14.9	37.79 \pm 15.2	32.86 \pm 15.3	43.27 \pm 15.3	38.17 \pm 15.9
Jeffreys CORAL	41.68 \pm 14.3	36.87 \pm 14.9	37.56 \pm 14.5	32.79 \pm 14.8	43.63 \pm 15.2	38.52 \pm 16.0
Stein CORAL	42.09 \pm 14.5	37.00 \pm 14.9	37.25 \pm 14.6	32.42 \pm 14.8	43.50 \pm 14.8	38.47 \pm 15.6
Linear MMCD	41.89 \pm 14.8	36.92 \pm 15.3	37.39 \pm 14.9	32.55 \pm 15.1	39.88 \pm 15.1	34.70 \pm 15.6
Kernelized MMCD	41.30 \pm 14.5	36.36 \pm 14.9	37.79 \pm 15.2	32.86 \pm 15.3	43.27 \pm 15.3	38.17 \pm 15.9
MMDA	46.16 \pm 15.3	41.27 \pm 16.0	41.10 \pm 16.0	36.93 \pm 16.1	43.76 \pm 15.1	38.77 \pm 15.9
DDC	42.08 \pm 14.5	37.21 \pm 15.0	37.05 \pm 14.6	32.45 \pm 14.8	43.07 \pm 15.3	37.86 \pm 16.2
Linear DAN	42.00 \pm 14.3	37.18 \pm 14.6	37.53 \pm 14.7	32.64 \pm 15.2	44.20 \pm 15.0	39.17 \pm 15.7
Squared DAN	42.47 \pm 15.0	37.65 \pm 15.5	36.98 \pm 14.1	32.10 \pm 14.2	43.37 \pm 14.8	38.16 \pm 15.9
HoMM _{p=3}	43.96 \pm 14.8	39.36 \pm 15.7	38.52 \pm 14.6	34.33 \pm 15.0	47.65 \pm 15.7	43.30 \pm 16.6
DANN	34.96 \pm 13.5	29.52 \pm 13.9	33.95 \pm 13.6	28.62 \pm 14.0	35.60 \pm 15.3	30.20 \pm 16.2
CDAN	40.83 \pm 14.1	35.35 \pm 14.8	37.90 \pm 13.3	32.38 \pm 14.1	19.86 \pm 16.1	13.30 \pm 16.5
DIRT-T	40.53 \pm 14.7	34.62 \pm 15.8	45.93 \pm 14.8	40.88 \pm 15.7	38.24 \pm 14.3	32.63 \pm 15.6
DSAN	7.24 \pm 1.8	1.0 \pm 0.3	6.91 \pm 1.8	1.03 \pm 0.4	7.36 \pm 1.8	0.99 \pm 0.3
CoDATS	33.79 \pm 13.4	28.07 \pm 14.2	37.79 \pm 14.3	33.00 \pm 14.8	35.17 \pm 15.0	29.58 \pm 15.7
AdvSKM	40.96 \pm 14.6	36.23 \pm 15.1	37.86 \pm 14.0	33.27 \pm 14.4	43.34 \pm 15.4	38.20 \pm 16.3
Sinkhorn	41.18 \pm 14.3	35.22 \pm 15.3	30.19 \pm 11.8	23.25 \pm 11.9	28.66 \pm 12.1	22.44 \pm 12.3

racy of 80.11% with a high standard deviation (18.3%) across different domain scenarios. The F1-scores are marginally lower than the accuracy, as the classes are approximately

Table 2.10: Epilepsy (Villar et al., 2016) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	98.12 \pm 1.2	98.17 \pm 1.1	97.19 \pm 1.8	97.34 \pm 1.7	68.75 \pm 2.2	69.05 \pm 2.4
CS	84.38 \pm 6.3	83.81 \pm 6.5	92.19 \pm 1.0	91.89 \pm 1.1	66.56 \pm 3.5	65.33 \pm 4.1
PC	99.06 \pm 1.2	99.09 \pm 1.2	99.38 \pm 0.8	99.40 \pm 0.7	77.81 \pm 3.2	77.31 \pm 3.1
KL	98.12 \pm 1.2	98.17 \pm 1.1	94.06 \pm 2.7	93.72 \pm 2.9	47.19 \pm 3.9	39.09 \pm 6.1
JSD	98.12 \pm 1.2	98.17 \pm 1.1	91.25 \pm 3.6	91.06 \pm 3.9	37.50 \pm 4.0	21.89 \pm 4.5
Linear MMD	98.12 \pm 1.2	98.17 \pm 1.1	98.75 \pm 0.6	98.79 \pm 0.6	75.00 \pm 1.4	74.91 \pm 1.6
kMMD	98.12 \pm 1.2	98.17 \pm 1.1	99.38 \pm 0.8	99.40 \pm 0.7	78.75 \pm 1.9	78.97 \pm 1.5
Deep CORAL	98.12 \pm 1.2	98.17 \pm 1.1	99.38 \pm 0.8	99.40 \pm 0.7	78.75 \pm 1.9	78.97 \pm 1.5
Jeffreys CORAL	98.12 \pm 1.2	98.17 \pm 1.1	99.38 \pm 0.8	99.40 \pm 0.7	78.44 \pm 1.5	78.67 \pm 1.2
Stein CORAL	98.12 \pm 1.2	98.17 \pm 1.1	99.38 \pm 0.8	99.40 \pm 0.7	78.44 \pm 1.5	78.67 \pm 1.2
Linear MMCD	98.12 \pm 1.2	98.17 \pm 1.1	98.75 \pm 0.6	98.79 \pm 0.6	75.00 \pm 3.0	75.08 \pm 2.9
Kernelized MMCD	98.12 \pm 1.2	98.17 \pm 1.1	99.38 \pm 0.8	99.40 \pm 0.7	78.75 \pm 1.9	78.97 \pm 1.5
MMDA	96.56 \pm 0.6	96.60 \pm 0.7	99.06 \pm 1.2	99.09 \pm 1.2	66.88 \pm 3.2	66.40 \pm 3.9
DDC	98.12 \pm 1.2	98.17 \pm 1.1	99.38 \pm 0.8	99.40 \pm 0.7	78.44 \pm 1.5	78.66 \pm 1.2
Linear DAN	98.75 \pm 1.2	98.79 \pm 1.1	98.75 \pm 0.6	98.79 \pm 0.6	74.38 \pm 2.5	74.45 \pm 13.0
Squared DAN	98.12 \pm 1.2	98.17 \pm 1.1	99.06 \pm 0.8	99.10 \pm 0.7	78.44 \pm 2.1	78.53 \pm 2.0
HoMM _{p=3}	92.19 \pm 2.6	91.57 \pm 3.2	99.06 \pm 0.8	99.10 \pm 0.7	65.62 \pm 2.6	63.81 \pm 3.3
DANN	96.88 \pm 2.0	96.77 \pm 2.2	95.31 \pm 2.6	95.27 \pm 2.8	68.44 \pm 4.8	67.28 \pm 5.4
CDAN	95.94 \pm 1.9	95.95 \pm 2.0	92.50 \pm 7.1	92.00 \pm 7.8	70.62 \pm 5.1	69.40 \pm 5.5
DIRT-T	88.75 \pm 2.5	87.94 \pm 2.8	90.00 \pm 1.2	89.52 \pm 1.3	80.00 \pm 2.7	75.47 \pm 2.2
DSAN	69.06 \pm 3.0	61.42 \pm 5.5	97.19 \pm 0.6	97.16 \pm 0.7	66.56 \pm 5.0	64.66 \pm 4.6
CoDATS	96.56 \pm 1.2	96.56 \pm 1.3	96.25 \pm 2.7	96.21 \pm 2.8	63.44 \pm 3.4	61.33 \pm 4.1
AdvSKM	98.12 \pm 1.2	98.10 \pm 1.2	98.75 \pm 0.6	98.79 \pm 0.6	77.81 \pm 3.2	77.68 \pm 2.8
Sinkhorn	96.25 \pm 0.8	96.28 \pm 0.8	90.31 \pm 4.6	89.72 \pm 5.2	64.69 \pm 4.4	63.84 \pm 4.5

Table 2.11: Face detection (Olivetti et al., 2014) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	60.12 \pm 2.3	58.22 \pm 4.2	51.63 \pm 1.5	48.06 \pm 5.7	64.86 \pm 0.8	64.84 \pm 0.8
CS	49.16 \pm 0.0	32.96 \pm 0.0	48.92 \pm 0.4	38.28 \pm 4.5	49.16 \pm 0.0	32.96 \pm 0.0
PC	59.59 \pm 2.0	58.22 \pm 3.2	52.62 \pm 1.1	47.76 \pm 3.7	65.43 \pm 0.8	65.41 \pm 0.8
KL	59.01 \pm 1.8	56.36 \pm 3.1	51.97 \pm 0.4	46.44 \pm 4.4	50.15 \pm 0.8	33.40 \pm 0.4
JSD	59.59 \pm 2.1	57.96 \pm 3.8	50.87 \pm 1.4	41.62 \pm 5.9	50.52 \pm 0.6	33.59 \pm 0.2
Linear MMD	63.20 \pm 1.2	63.06 \pm 1.3	53.31 \pm 1.1	52.74 \pm 1.5	64.66 \pm 0.8	64.58 \pm 0.8
kMMD	59.47 \pm 0.6	58.17 \pm 1.2	53.31 \pm 0.5	51.94 \pm 1.0	65.47 \pm 0.8	65.46 \pm 0.8
Deep CORAL	60.38 \pm 2.2	59.04 \pm 3.3	52.65 \pm 1.3	50.96 \pm 2.4	65.98 \pm 0.9	65.97 \pm 0.9
Jeffreys CORAL	60.46 \pm 3.0	58.60 \pm 5.5	52.21 \pm 0.9	50.25 \pm 2.2	64.53 \pm 1.9	64.52 \pm 1.8
Stein CORAL	59.42 \pm 2.1	57.15 \pm 4.0	52.71 \pm 1.0	51.26 \pm 2.4	65.85 \pm 0.8	65.85 \pm 0.8
Linear MMCD	62.29 \pm 1.8	61.36 \pm 3.1	52.76 \pm 2.0	49.23 \pm 7.7	65.50 \pm 1.5	65.47 \pm 1.5
Kernelized MMCD	60.38 \pm 2.2	59.04 \pm 3.3	52.65 \pm 1.3	50.69 \pm 2.4	65.98 \pm 0.9	65.97 \pm 0.9
MMDA	58.83 \pm 1.1	57.88 \pm 1.7	51.27 \pm 1.3	48.36 \pm 2.0	66.17 \pm 0.6	66.15 \pm 0.7
DDC	59.82 \pm 2.8	58.24 \pm 5.1	53.25 \pm 1.5	50.19 \pm 2.4	65.00 \pm 0.9	64.97 \pm 0.9
Linear DAN	61.10 \pm 2.3	59.76 \pm 3.7	53.32 \pm 1.0	51.18 \pm 3.9	64.95 \pm 0.9	64.93 \pm 0.9
Squared DAN	61.80 \pm 1.3	60.69 \pm 1.9	53.17 \pm 2.0	51.71 \pm 2.2	65.38 \pm 1.3	65.25 \pm 1.3
HoMM _{p=3}	58.58 \pm 2.9	54.55 \pm 6.2	51.80 \pm 1.4	47.17 \pm 4.9	67.87 \pm 0.7	67.85 \pm 0.7
DANN	60.96 \pm 1.0	60.31 \pm 0.9	52.29 \pm 1.3	50.81 \pm 1.7	65.69 \pm 1.2	65.67 \pm 1.2
CDAN	60.95 \pm 1.4	60.17 \pm 1.6	52.13 \pm 2.1	48.20 \pm 6.3	65.09 \pm 1.4	65.01 \pm 1.4
DIRT-T	57.47 \pm 2.8	50.90 \pm 7.1	52.36 \pm 2.7	44.34 \pm 8.9	62.80 \pm 2.3	62.57 \pm 2.3
DSAN	49.48 \pm 0.7	33.15 \pm 0.3	52.44 \pm 1.6	48.12 \pm 5.1	49.21 \pm 0.1	33.19 \pm 0.2
CoDATS	60.52 \pm 2.1	57.69 \pm 3.9	52.70 \pm 1.6	49.38 \pm 7.1	68.29 \pm 0.4	68.23 \pm 0.4
AdvSKM	59.27 \pm 1.6	58.18 \pm 2.9	53.31 \pm 0.8	50.70 \pm 4.2	65.43 \pm 1.8	65.31 \pm 2.0
Sinkhorn	61.16 \pm 1.9	59.61 \pm 2.8	52.85 \pm 1.3	50.27 \pm 3.0	59.85 \pm 7.2	53.13 \pm 15.4

distributed evenly (see Figure 2.1a). DIRT-T performs similarly well on the UCI HAR dataset, achieving an accuracy of 92.02% with a slightly lower F1-score (91.97%) due to

Table 2.12: PenDigits (Alimoglu & Alpaydin, 1997) dataset. **Bold** are best results.

Method	CNN		ResNet18		TCN	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	96.34 \pm 0.2	96.30 \pm 0.2	77.15 \pm 0.8	77.01 \pm 1.0	96.16 \pm 0.6	96.10 \pm 0.6
CS	96.05 \pm 0.4	95.97 \pm 0.4	75.83 \pm 1.1	75.73 \pm 1.1	95.67 \pm 1.6	94.56 \pm 0.6
PC	96.15 \pm 0.5	96.10 \pm 0.5	77.59 \pm 0.8	77.49 \pm 1.0	95.43 \pm 2.0	95.36 \pm 2.1
KL	95.81 \pm 0.8	95.76 \pm 0.9	63.97 \pm 4.8	60.31 \pm 6.4	10.42 \pm 0.6	1.89 \pm 0.1
JSD	95.82 \pm 0.7	95.77 \pm 0.7	56.37 \pm 5.4	54.03 \pm 5.1	10.01 \pm 0.6	1.82 \pm 0.1
Linear MMD	96.29 \pm 0.4	96.23 \pm 0.4	77.14 \pm 0.7	76.94 \pm 0.9	88.49 \pm 13.1	87.31 \pm 14.9
kMMD	95.91 \pm 0.6	95.83 \pm 0.6	77.54 \pm 0.5	77.40 \pm 0.9	95.46 \pm 0.5	95.42 \pm 0.5
Deep CORAL	96.04 \pm 0.7	95.98 \pm 0.7	77.49 \pm 0.7	77.37 \pm 0.9	96.16 \pm 0.4	96.12 \pm 0.5
Jeffreys CORAL	96.56 \pm 0.3	96.52 \pm 0.3	77.49 \pm 0.9	77.38 \pm 1.2	95.42 \pm 1.1	95.35 \pm 1.1
Stein CORAL	96.15 \pm 0.6	96.10 \pm 0.7	77.41 \pm 0.7	77.23 \pm 1.0	95.96 \pm 0.8	95.92 \pm 0.8
Linear MMCD	96.13 \pm 0.4	96.08 \pm 0.5	76.98 \pm 0.8	76.87 \pm 1.0	84.51 \pm 6.3	81.55 \pm 7.7
Kernelized MMCD	96.04 \pm 0.7	95.98 \pm 0.7	77.49 \pm 0.7	77.37 \pm 0.9	96.16 \pm 0.4	96.12 \pm 0.5
MMDA	95.72 \pm 0.6	95.59 \pm 0.7	78.53 \pm 1.2	78.50 \pm 1.3	96.26 \pm 0.6	96.22 \pm 0.7
DDC	95.83 \pm 0.4	95.79 \pm 0.4	77.32 \pm 0.5	77.16 \pm 0.8	95.64 \pm 0.5	95.59 \pm 0.6
Linear DAN	96.08 \pm 0.5	96.01 \pm 0.6	77.14 \pm 0.5	77.01 \pm 0.8	95.89 \pm 0.6	95.84 \pm 0.6
Squared DAN	96.20 \pm 0.5	96.15 \pm 0.5	77.37 \pm 0.6	77.27 \pm 0.8	95.83 \pm 1.0	95.78 \pm 1.0
HoMM _{p=3}	96.29 \pm 0.5	96.22 \pm 0.6	78.46 \pm 0.7	78.36 \pm 0.8	95.52 \pm 0.6	95.49 \pm 0.6
DANN	96.65 \pm 0.3	96.65 \pm 0.3	76.48 \pm 1.8	76.20 \pm 1.6	94.54 \pm 0.8	94.47 \pm 0.8
CDAN	96.43 \pm 0.4	96.41 \pm 0.5	75.48 \pm 1.9	75.25 \pm 2.0	92.25 \pm 5.7	91.36 \pm 7.3
DIRT-T	96.67 \pm 0.2	96.67 \pm 0.2	79.21 \pm 0.5	78.86 \pm 0.7	96.99 \pm 0.3	96.99 \pm 0.3
DSAN	11.07 \pm 0.0	1.99 \pm 0.0	9.95 \pm 0.8	1.87 \pm 0.1	95.83 \pm 2.3	95.62 \pm 2.7
CoDATS	96.24 \pm 0.4	96.20 \pm 0.4	83.68 \pm 1.3	83.37 \pm 1.3	94.21 \pm 2.0	93.96 \pm 2.3
AdvSKM	96.46 \pm 0.2	96.40 \pm 0.2	77.43 \pm 0.8	77.05 \pm 1.0	95.85 \pm 0.8	95.80 \pm 0.9
Sinkhorn	96.46 \pm 0.3	96.43 \pm 0.3	76.29 \pm 0.9	76.08 \pm 1.1	72.07 \pm 6.4	67.48 \pm 8.1

equal class distributions (see Figure 2.1b). CDAN and DSAN also have high accuracy results on all datasets. However, on the WISDM dataset, the standard deviation between different domain scenarios is high due to unequal contributions from participants, and the F1-score is lower due to the predominant classes 1 and 5 (see Figure 2.1c). DANN achieves the highest accuracy of 63.37% using the CNN encoder, while lower results are obtained using the ResNet18 encoder (59.77% of DSAN) and the TCN encoder (60.58% of squared DAN). All DA methods achieve similarly high results on the SSC (EEG) dataset with the CNN encoder, and DIRT-T (73.48%) and DANN (73.09%) outperform other DA methods. The results drop notably for the ResNet18 and TCN encoders. The classification task on the uWave dataset is less challenging, and hence, classification results are higher, e.g., CDAN and DIRT-T achieve 96.43% accuracy (see Table 2.7). DSAN fails to classify the dataset. While the TCN encoder perpetuates high accuracies (e.g., 89.90% of HoMM_{p=3}), the ResNet18 encoder reduces accuracies to between 49.90% to 59.25%. As each participant contributed equally (see Figure 2.1e), the standard deviation is low, and the F1-scores are marginally lower, although the class 2 is predominant. The linear MMCD method outperforms or is on par with the kernelized MMCD method. The squared version of DAN marginally outperforms the linear version for all datasets. In Table 2.8, we present the results for the DA methods on the univariate finger movements dataset. The Cosine similarity method, as standard loss for CMR, achieves high accuracies (53.27% on average) when used with the CNN encoder. On the other hand, Sinkhorn outperforms all methods, achieving an average accuracy of 55.07% when used with the TCN encoder. Although HoMM_{p=3} achieves the highest accuracy (47.65%) on the gestures mid air dataset

Table 2.13: DA results in % for the OnHW datasets based on AdaTime. Results show mean and standard deviation over five runs for the OnHW-symbols and split OnHW-equations datasets, and over five cross-validation splits and five runs for the OnHW-chars dataset. CNN as encoder network. **Bold** are best results. Comparison to the results of the optimal transport (OT) method based on Sinkhorn transport from Ott et al. (2022a).

Method	OnHW-symbols (L→R)		OnHW-symbols (R→L)		Split OnHW-equations (L→R)	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	77.81 ± 4.3	78.56 ± 5.5	79.33 ± 3.0	78.96 ± 3.3	85.29 ± 1.7	84.39 ± 1.8
CS	14.69 ± 3.8	9.48 ± 2.8	6.88 ± 0.8	2.83 ± 1.0	11.60 ± 3.6	5.02 ± 2.9
PC	77.19 ± 2.8	77.08 ± 5.7	80.13 ± 2.7	79.46 ± 3.4	78.96 ± 6.6	76.81 ± 7.3
KL	78.12 ± 3.7	79.16 ± 4.3	80.67 ± 4.1	80.55 ± 4.7	84.46 ± 3.5	83.07 ± 3.9
JSD	76.88 ± 4.0	77.18 ± 5.8	78.84 ± 3.1	78.56 ± 3.6	10.74 ± 6.2	6.34 ± 4.8
Linear MMD	76.56 ± 2.9	75.88 ± 3.2	81.79 ± 2.1	81.41 ± 2.7	88.33 ± 0.9	87.78 ± 1.3
kMMD	78.44 ± 2.0	79.75 ± 1.7	79.78 ± 3.2	79.39 ± 3.8	87.00 ± 0.6	86.25 ± 0.8
Deep CORAL	84.23 ± 1.0	83.94 ± 0.9	77.73 ± 5.9	77.72 ± 5.6	84.66 ± 1.7	83.61 ± 1.9
Jeffreys CORAL	77.81 ± 2.6	78.74 ± 2.6	79.64 ± 1.8	79.43 ± 2.1	86.84 ± 1.7	85.97 ± 1.9
Stein CORAL	76.25 ± 2.3	77.63 ± 2.3	80.18 ± 1.2	79.99 ± 1.7	85.99 ± 2.1	85.16 ± 2.5
Linear MMCD	76.25 ± 2.6	76.79 ± 3.6	81.29 ± 4.0	81.14 ± 4.3	88.37 ± 1.5	87.59 ± 1.9
Kernelized MMCD	76.25 ± 2.6	76.79 ± 3.6	80.45 ± 2.7	80.16 ± 3.0	87.48 ± 1.4	86.75 ± 1.6
MMDA	77.23 ± 2.5	77.14 ± 2.5	61.86 ± 5.8	61.82 ± 5.4	75.95 ± 4.4	74.95 ± 4.5
DDC	84.27 ± 2.3	83.85 ± 2.6	77.32 ± 5.9	77.24 ± 5.7	84.33 ± 1.9	83.14 ± 2.2
Linear DAN	76.56 ± 2.7	76.26 ± 2.5	81.83 ± 3.3	81.39 ± 4.2	88.37 ± 1.4	87.92 ± 1.5
Squared DAN	77.81 ± 10.4	78.15 ± 12.4	80.62 ± 2.6	80.20 ± 3.5	87.77 ± 2.7	86.91 ± 2.9
HoMM _{p=3}	83.33 ± 2.9	83.99 ± 2.1	76.38 ± 3.6	76.51 ± 3.2	83.87 ± 3.7	83.23 ± 3.6
DANN	77.07 ± 6.2	76.90 ± 6.6	74.96 ± 6.2	73.69 ± 7.7	71.62 ± 6.8	71.49 ± 6.0
CDAN	65.22 ± 10.3	63.87 ± 12.0	72.37 ± 4.5	71.93 ± 5.1	52.90 ± 20.9	52.75 ± 22.0
DIRT-T	76.88 ± 3.7	77.23 ± 3.2	58.84 ± 5.3	59.40 ± 4.8	82.71 ± 1.9	81.17 ± 2.3
DSAN	6.18 ± 1.2	1.53 ± 1.8	6.25 ± 0.4	0.88 ± 0.2	7.96 ± 0.0	0.98 ± 0.0
CoDATS	80.63 ± 3.1	80.71 ± 2.6	67.01 ± 6.5	66.75 ± 6.3	76.23 ± 4.5	75.73 ± 4.7
AdvSKM	84.08 ± 3.5	84.38 ± 3.4	81.03 ± 2.4	80.57 ± 3.0	83.99 ± 2.9	83.08 ± 3.4
Sinkhorn	81.56 ± 2.0	82.77 ± 1.8	78.93 ± 4.1	78.47 ± 4.2	85.67 ± 1.4	84.33 ± 2.1
OT ¹ [kMMD]	85.09 ± 7.7	-	-	-	84.03 ± 9.4	-
OT ¹ [HoMM _{p=3}]	70.03 ± 7.4	-	-	-	62.37 ± 13.3	-
OT ¹ [CORAL (J)]	80.92 ± 8.0	-	-	-	82.24 ± 10.3	-
OT ¹ [CORAL (S)]	82.31 ± 7.8	-	-	-	76.12 ± 13.0	-
Method	Split OnHW-equations (R→L)		OnHW-chars (combined) (L→R)		OnHW-chars (combined) (R→L)	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	92.76 ± 0.8	92.35 ± 0.8	75.51 ± 3.4	73.75 ± 4.6	64.02 ± 0.2	62.52 ± 0.3
CS	20.82 ± 19.7	16.63 ± 14.6	5.59 ± 4.1	3.72 ± 3.8	1.94 ± 0.0	0.07 ± 0.0
PC	92.56 ± 1.2	92.07 ± 1.3	75.17 ± 2.0	73.48 ± 3.0	63.85 ± 0.2	62.40 ± 0.3
KL	92.44 ± 1.6	91.92 ± 1.9	74.01 ± 3.1	72.30 ± 3.9	63.72 ± 0.3	62.13 ± 0.3
JSD	92.42 ± 1.0	91.94 ± 1.1	74.22 ± 3.4	72.48 ± 4.2	63.45 ± 0.1	61.85 ± 0.1
Linear MMD	92.67 ± 0.4	92.14 ± 0.5	74.39 ± 3.3	73.60 ± 3.8	63.41 ± 0.1	61.82 ± 0.1
kMMD	92.62 ± 0.9	92.14 ± 1.0	74.92 ± 2.8	73.40 ± 3.5	64.05 ± 0.0	62.55 ± 0.0
Deep CORAL	92.27 ± 1.1	91.62 ± 1.1	75.39 ± 3.2	73.70 ± 3.7	64.98 ± 0.2	63.55 ± 0.3
Jeffreys CORAL	92.57 ± 0.6	92.08 ± 0.5	74.51 ± 2.9	72.54 ± 3.5	63.72 ± 0.3	62.22 ± 0.3
Stein CORAL	92.72 ± 0.7	92.28 ± 0.8	74.62 ± 2.9	73.08 ± 3.6	63.90 ± 0.2	62.37 ± 0.2
Linear MMCD	93.10 ± 0.6	92.59 ± 0.5	75.12 ± 2.8	73.61 ± 3.4	63.71 ± 0.1	62.22 ± 0.1
Kernelized MMCD	92.68 ± 1.2	92.24 ± 1.4	74.96 ± 2.7	73.61 ± 3.4	63.72 ± 0.3	62.26 ± 0.3
MMDA	92.08 ± 0.6	91.24 ± 0.7	62.98 ± 2.7	61.43 ± 3.3	57.64 ± 0.2	56.39 ± 0.2
DDC	92.60 ± 0.9	92.00 ± 0.9	71.30 ± 2.5	70.93 ± 2.3	65.17 ± 0.1	63.80 ± 0.1
Linear DAN	92.62 ± 0.9	92.23 ± 1.0	74.90 ± 3.0	72.93 ± 4.0	63.65 ± 0.2	62.16 ± 0.2
Squared DAN	92.75 ± 0.7	92.34 ± 0.7	75.14 ± 2.7	73.46 ± 3.4	63.96 ± 0.2	62.50 ± 0.1
HoMM _{p=3}	93.08 ± 0.9	92.55 ± 0.8	78.14 ± 2.1	76.49 ± 2.8	63.39 ± 0.1	62.34 ± 0.1
DANN	91.52 ± 0.4	90.83 ± 0.6	71.70 ± 3.3	69.53 ± 4.4	62.60 ± 0.5	61.08 ± 0.4
CDAN	89.23 ± 1.9	88.42 ± 2.3	66.27 ± 4.2	66.40 ± 4.0	45.29 ± 8.5	46.39 ± 8.0
DIRT-T	91.53 ± 0.8	90.88 ± 0.9	74.88 ± 1.6	72.90 ± 2.7	57.78 ± 0.1	56.57 ± 0.1
DSAN	8.38 ± 0.0	1.03 ± 0.0	0.60 ± 0.1	0.03 ± 0.0	1.80 ± 0.0	0.07 ± 0.0
CoDATS	91.85 ± 1.0	91.22 ± 1.0	71.07 ± 4.1	69.11 ± 4.6	57.34 ± 0.7	55.49 ± 0.7
AdvSKM	92.33 ± 0.5	91.83 ± 0.4	74.40 ± 3.4	72.59 ± 4.5	64.98 ± 0.0	63.51 ± 0.0
Sinkhorn	92.91 ± 1.3	92.34 ± 1.5	76.96 ± 2.5	75.27 ± 3.4	64.06 ± 0.1	62.79 ± 0.1
OT ¹ [kMMD]	-	-	73.87 ± 11.1	-	-	-
OT ¹ [HoMM _{p=3}]	-	-	56.84 ± 9.6	-	-	-
OT ¹ [CORAL (J)]	-	-	72.30 ± 9.1	-	-	-
OT ¹ [CORAL (S)]	-	-	66.50 ± 7.7	-	-	-

¹ OT is trained for one cross-validation split for OnHW-symbols and split OnHW-equations (mean over all samples), and trained and averaged over five cross-validation splits for OnHW-chars.

as shown in Table 2.9, it fails significantly short of the target-only accuracy of 69.92% as indicated in Table 2.2. Several DA methods, including PC, Deep CORAL, Jeffreys CORAL, and Stein CORAL, perform similarly well (99.38%) on the epilepsy dataset, as

Table 2.14: DA results in % for the OnHW datasets based on AdaTime. ResNet18 as encoder network. Results show mean and standard deviation over five runs for the OnHW-symbols and split OnHW-equations datasets and over five cross-validation splits and five runs for the OnHW-chars dataset. **Bold** are best results.

Method	OnHW-symbols (L→R)		OnHW-symbols (R→L)		Split OnHW-equations (L→R)	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	60.31 ± 1.8	59.01 ± 4.3	49.15 ± 8.0	48.06 ± 5.5	57.53 ± 7.4	55.07 ± 7.5
CS	40.62 ± 7.2	36.81 ± 9.6	33.93 ± 8.0	29.79 ± 8.6	9.66 ± 2.5	3.72 ± 2.2
PC	55.94 ± 1.7	53.42 ± 1.8	51.70 ± 8.1	49.87 ± 7.3	59.36 ± 3.4	58.09 ± 3.4
KL	59.38 ± 5.3	57.34 ± 6.9	46.16 ± 8.9	43.44 ± 8.1	22.60 ± 9.6	17.86 ± 8.2
JSD	44.69 ± 5.9	40.53 ± 6.5	37.23 ± 10.8	33.88 ± 9.0	9.63 ± 3.3	3.88 ± 2.0
Linear MMD	61.88 ± 8.5	60.61 ± 7.7	52.77 ± 6.0	50.82 ± 5.4	70.08 ± 3.5	69.93 ± 3.2
kMMD	61.25 ± 3.4	59.33 ± 3.9	51.34 ± 7.8	49.31 ± 6.8	59.97 ± 3.5	58.29 ± 4.2
Deep CORAL	78.36 ± 1.1	78.50 ± 1.6	51.92 ± 7.5	50.21 ± 6.6	72.99 ± 2.2	71.91 ± 2.7
Jeffreys CORAL	58.44 ± 5.5	55.62 ± 6.0	51.52 ± 8.0	49.76 ± 6.7	59.10 ± 4.3	58.65 ± 3.9
Stein CORAL	59.06 ± 3.7	57.73 ± 4.4	51.74 ± 7.1	49.95 ± 5.9	58.81 ± 2.3	57.53 ± 3.4
Linear MMCD	61.25 ± 5.2	60.28 ± 7.1	53.39 ± 5.8	51.60 ± 5.9	68.96 ± 3.9	69.27 ± 3.7
Kernelized MMCD	56.56 ± 3.2	55.83 ± 6.8	51.92 ± 7.5	50.21 ± 6.6	59.49 ± 5.9	57.76 ± 5.8
MMDA	65.61 ± 6.3	65.29 ± 6.3	41.25 ± 4.9	40.17 ± 3.8	67.87 ± 4.1	65.73 ± 4.1
DDC	77.98 ± 3.1	78.15 ± 3.2	51.88 ± 6.7	50.06 ± 5.3	71.86 ± 4.5	70.00 ± 4.9
Linear DAN	56.88 ± 6.5	53.28 ± 7.1	53.17 ± 5.8	51.39 ± 4.7	67.72 ± 3.9	67.59 ± 3.1
Squared DAN	61.25 ± 5.6	59.66 ± 5.5	51.52 ± 7.7	49.69 ± 6.2	62.92 ± 4.2	62.28 ± 3.8
HoMM _{p=3}	64.83 ± 3.8	65.59 ± 3.5	39.38 ± 4.7	39.72 ± 3.8	71.44 ± 8.3	69.31 ± 8.7
DANN	63.03 ± 12.6	60.36 ± 12.9	50.54 ± 4.7	49.25 ± 4.6	54.38 ± 5.3	54.06 ± 4.8
CDAN	50.98 ± 7.5	49.24 ± 8.9	38.79 ± 13.9	35.04 ± 15.8	45.72 ± 9.9	43.87 ± 8.8
DIRT-T	53.96 ± 8.7	53.73 ± 10.1	39.06 ± 3.6	38.07 ± 3.9	59.50 ± 7.6	58.62 ± 7.7
DSAN	44.41 ± 8.8	42.86 ± 8.3	21.70 ± 3.4	17.00 ± 2.4	8.66 ± 0.8	1.86 ± 0.5
CoDATS	64.83 ± 8.3	65.28 ± 8.6	49.91 ± 1.7	47.68 ± 4.1	54.08 ± 3.8	53.93 ± 3.3
AdvSKM	77.31 ± 2.5	76.61 ± 2.6	56.03 ± 8.2	54.55 ± 9.2	72.76 ± 2.6	71.57 ± 2.6
Sinkhorn	45.31 ± 5.5	42.36 ± 5.4	30.31 ± 9.0	25.20 ± 8.8	70.85 ± 5.3	70.58 ± 4.9
Method	Split OnHW-equations (R→L)		OnHW-chars (combined) (L→R)		OnHW-chars (combined) (R→L)	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	79.47 ± 3.7	79.76 ± 3.6	46.18 ± 3.8	44.31 ± 4.0	32.40 ± 0.5	29.86 ± 0.5
CS	32.81 ± 24.7	29.65 ± 27.1	13.32 ± 1.7	11.03 ± 1.1	3.78 ± 0.4	0.99 ± 0.3
PC	79.51 ± 1.6	79.85 ± 1.4	46.32 ± 3.5	44.32 ± 3.7	31.03 ± 0.2	28.75 ± 0.2
KL	44.10 ± 5.1	40.38 ± 5.1	44.88 ± 3.5	42.55 ± 4.0	31.67 ± 0.2	29.24 ± 0.2
JSD	19.95 ± 3.4	15.97 ± 2.0	6.64 ± 0.6	3.39 ± 0.7	23.17 ± 2.5	20.79 ± 2.7
Linear MMD	79.53 ± 1.7	79.79 ± 1.4	46.15 ± 4.8	44.45 ± 4.5	31.72 ± 0.6	29.39 ± 0.7
kMMD	79.51 ± 2.0	79.90 ± 1.5	46.51 ± 4.8	44.69 ± 5.1	31.68 ± 0.1	29.35 ± 0.2
Deep CORAL	79.09 ± 1.9	79.56 ± 1.5	63.98 ± 3.9	62.39 ± 4.5	31.68 ± 0.1	29.35 ± 0.2
Jeffreys CORAL	79.48 ± 2.2	79.92 ± 1.7	46.08 ± 4.9	44.42 ± 4.9	31.31 ± 0.2	29.08 ± 0.2
Stein CORAL	79.03 ± 1.7	79.38 ± 1.6	46.04 ± 4.3	44.08 ± 4.4	31.34 ± 0.4	29.05 ± 0.4
Linear MMCD	79.21 ± 2.2	79.32 ± 2.1	46.05 ± 4.9	44.15 ± 5.3	31.72 ± 0.6	29.39 ± 0.7
Kernelized MMCD	79.09 ± 1.9	79.56 ± 1.5	45.54 ± 4.3	43.53 ± 4.5	31.68 ± 0.1	29.35 ± 0.2
MMDA	75.50 ± 3.4	75.87 ± 3.0	56.61 ± 4.1	55.15 ± 4.5	29.29 ± 0.3	26.32 ± 0.3
DDC	79.42 ± 2.5	79.59 ± 2.2	63.31 ± 3.7	61.86 ± 4.3	31.43 ± 0.1	29.09 ± 0.1
Linear DAN	79.33 ± 2.8	79.55 ± 2.7	45.40 ± 4.0	43.44 ± 4.2	32.21 ± 0.3	29.51 ± 0.2
Squared DAN	79.88 ± 2.5	80.17 ± 2.3	46.59 ± 4.4	44.70 ± 4.6	31.80 ± 0.2	29.51 ± 0.2
HoMM _{p=3}	76.59 ± 2.4	76.93 ± 2.3	61.33 ± 4.2	59.51 ± 5.0	24.86 ± 0.8	21.47 ± 0.8
DANN	75.22 ± 2.0	74.77 ± 2.6	57.09 ± 4.0	55.64 ± 4.4	29.09 ± 0.8	27.14 ± 0.8
CDAN	55.88 ± 5.4	52.91 ± 7.2	52.63 ± 4.8	50.76 ± 4.5	15.09 ± 0.6	11.23 ± 0.6
DIRT-T	70.99 ± 1.8	70.52 ± 1.3	55.07 ± 4.3	53.33 ± 4.5	30.05 ± 0.2	27.64 ± 0.2
DSAN	8.85 ± 0.5	1.81 ± 0.6	0.89 ± 0.0	0.13 ± 0.1	1.97 ± 0.0	0.14 ± 0.0
CoDATS	76.13 ± 4.0	75.87 ± 3.7	61.30 ± 3.0	59.56 ± 4.0	29.24 ± 0.4	26.90 ± 0.4
AdvSKM	79.53 ± 1.7	79.75 ± 1.9	63.44 ± 3.7	61.90 ± 4.4	32.17 ± 0.1	29.98 ± 0.1
Sinkhorn	77.24 ± 3.3	77.45 ± 3.1	37.21 ± 3.8	34.48 ± 4.0	25.74 ± 1.5	22.54 ± 1.6

shown in Table 2.10. On the human activity datasets, linear MMD outperforms kMMD, especially on the face detection dataset, as demonstrated in Table 2.11. Additionally, as shown in Table 2.12, DIRT-T achieves the highest accuracy of 96.67% on the PenDigits dataset. While linear MMD outperforms kMMD, the variants Jeffreys and Stein CORAL outperform the standard Deep CORAL.

Evaluation of DA & Encoder Networks on the OnHW Datasets. Refer to Table 2.13, Table 2.14, and Table 2.15 for the results obtained from the OnHW-symbols, split OnHW-equations, and OnHW-chars (combined and writer-dependent) datasets. No-

Table 2.15: DA results in % for the OnHW datasets based on AdaTime. Results show mean and standard deviation over five runs for the OnHW-symbols and split OnHW-equations datasets and over five cross-validation splits and five runs for the OnHW-chars dataset. TCN as encoder network. **Bold** are best results.

Method	OnHW-symbols (L→R)		OnHW-symbols (R→L)		Split OnHW-equations (L→R)	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	6.25 ± 0.0	0.78 ± 0.0	5.62 ± 0.3	0.71 ± 0.0	8.01 ± 0.0	0.99 ± 0.0
CS	6.25 ± 0.0	0.78 ± 0.0	5.58 ± 0.0	0.70 ± 0.0	7.05 ± 0.0	0.88 ± 0.0
PC	6.25 ± 0.0	0.78 ± 0.0	5.58 ± 0.0	0.70 ± 0.0	7.05 ± 0.0	0.88 ± 0.0
KL	8.75 ± 3.0	1.06 ± 0.3	5.62 ± 0.4	0.71 ± 0.0	6.03 ± 2.3	0.75 ± 0.3
JSD	6.25 ± 0.0	0.78 ± 0.0	6.96 ± 2.5	0.86 ± 0.3	7.76 ± 2.1	0.96 ± 0.2
Linear MMD	5.62 ± 1.4	0.71 ± 0.2	5.58 ± 0.3	0.71 ± 0.0	8.01 ± 0.0	0.99 ± 0.0
kMMD	6.25 ± 0.0	0.79 ± 0.0	5.49 ± 0.2	0.69 ± 0.0	8.03 ± 0.0	1.01 ± 0.0
Deep CORAL	6.88 ± 1.8	2.91 ± 2.4	7.68 ± 2.2	4.03 ± 3.4	8.01 ± 0.0	0.99 ± 0.0
Jeffreys CORAL	6.25 ± 0.0	0.78 ± 0.0	5.45 ± 0.1	0.69 ± 0.0	8.01 ± 0.0	0.99 ± 0.0
Stein CORAL	6.25 ± 0.0	0.78 ± 0.0	5.45 ± 0.3	0.69 ± 0.0	8.01 ± 0.0	0.99 ± 0.0
Linear MMCD	5.94 ± 0.7	0.75 ± 0.1	5.62 ± 0.5	0.71 ± 0.1	8.01 ± 0.0	0.99 ± 0.0
Kernelized MMCD	6.88 ± 1.8	2.91 ± 2.4	7.68 ± 2.2	4.03 ± 3.4	8.01 ± 0.0	0.99 ± 0.0
MMDA	7.19 ± 3.4	6.62 ± 3.8	7.72 ± 1.8	7.34 ± 1.4	9.07 ± 0.4	2.23 ± 0.5
DDC	6.25 ± 0.0	0.78 ± 0.0	5.49 ± 0.2	0.69 ± 0.0	8.03 ± 0.0	1.01 ± 0.1
Linear DAN	6.25 ± 0.0	0.79 ± 0.0	5.49 ± 0.2	0.69 ± 0.0	7.82 ± 0.4	0.97 ± 0.0
Squared DAN	6.25 ± 0.0	0.78 ± 0.0	5.49 ± 0.2	0.78 ± 0.2	7.44 ± 0.5	0.92 ± 0.1
HoMM _{p=3}	8.44 ± 4.5	6.94 ± 4.3	7.99 ± 1.6	6.84 ± 1.4	10.29 ± 1.7	3.32 ± 0.9
DANN	6.25 ± 0.0	0.78 ± 0.0	5.76 ± 0.4	0.73 ± 0.1	9.26 ± 1.0	2.37 ± 1.5
CDAN	5.00 ± 0.7	0.63 ± 0.1	6.52 ± 2.4	0.81 ± 0.3	8.01 ± 0.0	0.99 ± 0.0
DIRT-T	16.56 ± 4.2	12.12 ± 4.8	14.15 ± 2.3	11.14 ± 2.4	45.45 ± 6.1	43.25 ± 6.5
DSAN	18.44 ± 1.3	14.84 ± 1.0	16.61 ± 1.9	11.91 ± 2.0	8.80 ± 1.7	1.80 ± 1.6
CoDATS	24.38 ± 3.6	22.26 ± 3.9	8.66 ± 4.6	4.21 ± 5.9	27.58 ± 1.6	23.68 ± 2.0
AdvSKM	6.25 ± 0.0	0.79 ± 0.0	5.71 ± 0.2	0.72 ± 0.0	8.01 ± 0.0	0.99 ± 0.0
Sinkhorn	6.25 ± 1.1	1.17 ± 1.0	7.32 ± 2.8	3.07 ± 1.4	5.95 ± 2.2	1.39 ± 0.8
Method	Split OnHW-equations (R→L)		OnHW-chars (combined) (L→R)		OnHW-chars (combined) (R→L)	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
MSE	8.37 ± 0.4	1.03 ± 0.0	0.58 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
CS	5.73 ± 0.0	0.72 ± 0.0	2.01 ± 0.6	0.08 ± 0.0	1.94 ± 0.0	0.07 ± 0.0
PC	5.73 ± 0.0	0.72 ± 0.0	2.01 ± 0.6	0.08 ± 0.0	1.94 ± 0.0	0.07 ± 0.0
KL	7.52 ± 1.9	0.93 ± 0.2	1.75 ± 0.2	0.07 ± 0.0	1.91 ± 0.0	0.07 ± 0.0
JSD	6.66 ± 2.2	0.83 ± 0.3	1.73 ± 0.1	0.07 ± 0.0	1.89 ± 0.0	0.07 ± 0.0
Linear MMD	8.37 ± 0.4	1.03 ± 0.0	0.58 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
kMMD	8.37 ± 0.4	1.03 ± 0.0	0.58 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
Deep CORAL	9.73 ± 2.1	2.04 ± 1.4	0.62 ± 0.1	0.06 ± 0.0	1.83 ± 0.0	0.08 ± 0.0
Jeffreys CORAL	8.37 ± 0.4	1.03 ± 0.0	0.6 ± 0.1	0.04 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
Stein CORAL	8.37 ± 0.4	1.03 ± 0.0	0.58 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
Linear MMCD	8.37 ± 0.4	1.03 ± 0.0	0.62 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
Kernelized MMCD	9.73 ± 2.1	2.04 ± 1.4	0.62 ± 0.1	0.06 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
MMDA	8.99 ± 1.0	2.60 ± 0.6	1.12 ± 0.5	0.27 ± 0.1	2.03 ± 0.1	0.28 ± 0.0
DDC	8.37 ± 0.4	1.03 ± 0.0	0.58 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
Linear DAN	8.37 ± 0.4	1.03 ± 0.0	0.74 ± 0.1	0.03 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
Squared DAN	7.32 ± 1.4	0.91 ± 0.2	0.58 ± 0.1	0.02 ± 0.0	1.83 ± 0.0	0.07 ± 0.0
HoMM _{p=3}	8.62 ± 0.3	1.79 ± 0.5	1.94 ± 0.4	0.73 ± 0.2	2.46 ± 0.1	0.72 ± 0.1
DANN	14.21 ± 12.1	8.08 ± 15.2	1.71 ± 1.0	1.25 ± 0.9	1.91 ± 0.0	0.08 ± 0.0
CDAN	8.38 ± 0.0	1.03 ± 0.0	1.30 ± 0.3	0.05 ± 0.0	1.93 ± 0.0	0.07 ± 0.0
DIRT-T	53.94 ± 1.4	52.02 ± 3.2	1.06 ± 0.1	0.57 ± 0.1	1.86 ± 0.0	0.09 ± 0.0
DSAN	7.92 ± 1.2	1.10 ± 0.2	0.73 ± 0.2	0.09 ± 0.1	1.79 ± 0.0	0.08 ± 0.0
CoDATS	33.79 ± 10.1	27.73 ± 10.2	0.71 ± 0.3	0.04 ± 0.0	1.85 ± 0.1	0.10 ± 0.0
AdvSKM	8.15 ± 0.3	1.01 ± 0.0	0.58 ± 0.5	0.02 ± 0.0	1.85 ± 0.0	0.07 ± 0.0
Sinkhorn	7.31 ± 1.8	1.27 ± 0.5	1.94 ± 0.4	0.26 ± 0.1	2.06 ± 0.1	0.43 ± 0.1

tably, the CNN encoder outperforms the ResNet18 encoder by extracting features that lead to high classification accuracy, whereas the TCN encoder fails to extract meaningful features. Moreover, when adapting left-handed writers to right-handed writers (L→R), the OnHW-symbols and OnHW-chars datasets yield higher accuracies than when adapting right-handed writers to left-handed writers (R→L). This is likely due to the right-handed writers datasets having more training samples than the left-handed writers datasets, as illustrated in Figure 2.2c and Figure 2.2e). In the case of split OnHW-equations dataset, both left-handed and right-handed writers contributed significantly to the samples (still with a predominance of right-handed writers), as illustrated in Figure 2.2d, hence the effect

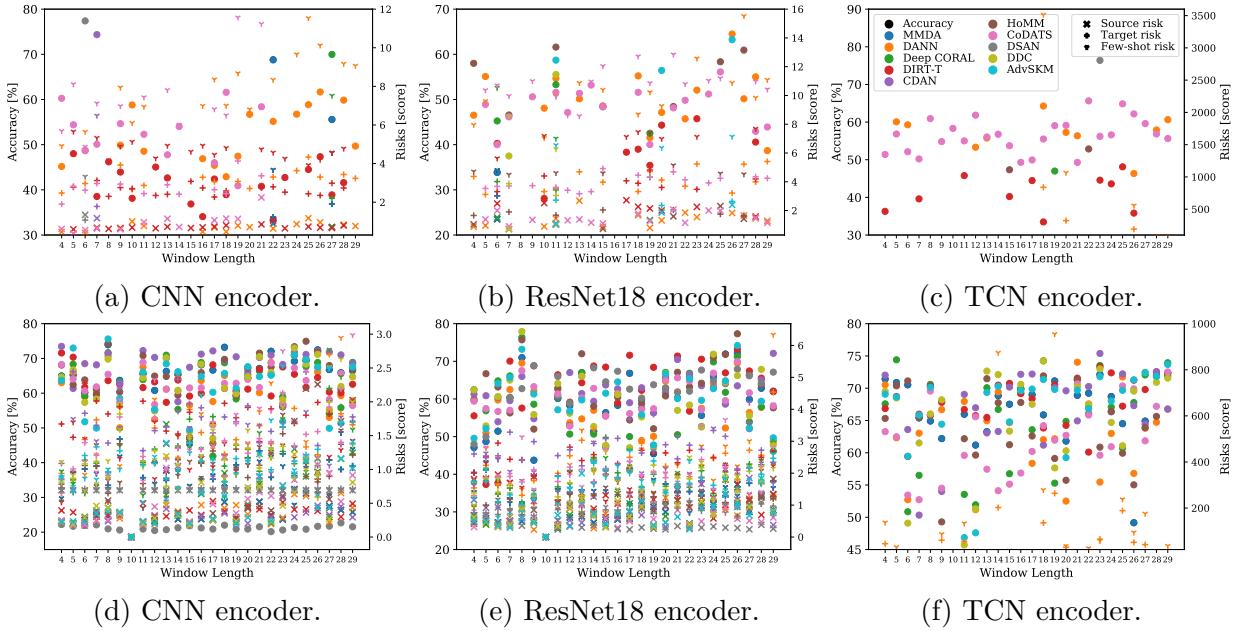


Figure 2.3: DA results in % for the GNSS-based dataset utilizing AdaTime for three encoder networks and for window lengths between 4 and 29. Results are averaged over five runs. Top: classification task with nine classes (one class for non-interference and eight classes for interference). Bottom: binary classification task (interference or non-interference).

of writer adaptation is not present. On the OnHW-symbols dataset, DDC (84.27%), Deep CORAL (84.23%), and AdvSKM (84.08%) were found to have the highest classification accuracy among all DA methods. Conversely, on the split OnHW-equations dataset, linear DAN (88.37%), linear MMCD (88.37%), and linear MMD (88.33%) outperforms all DA methods, with linearized versions being preferred over the kernelized versions. For the OnHW-chars dataset, $\text{HoMM}_{p=3}$ (78.14%), Sinkhorn (76.96%), and Deep CORAL (75.39%) yield the highest accuracies. While the optimal transport (OT) method by Ott et al. (2022a) is supervised and uses a different CNN encoder, a direct comparison to the remaining DA methods shows that using kMMD as similarity technique for feature embedding transformation results in the highest accuracy of 85.09% on the OnHW-symbols dataset. Notably, kMMD was found to be the preferred method over HoMM and Jeffreys and Stein CORAL for all OnHW datasets.

Evaluation of DA & Encoder Networks on the GNSS-based Dataset. An overview of classification results for GNSS-based interference detection is presented in Figure 2.3. Figure 2.3a through Figure 2.3c depict the classification results for the multi-class task consisting of eight interference classes. On the other hand, Figure 2.3d to Figure 2.3f demonstrate the results of the binary classification task. It is important to note that some DA methods failed to train properly, rendering them out-of-scale in the figures. In the case

of the multi-class problem, the CNN encoder outperforms the ResNet18 and TCN encoders, as observed in previous datasets. The CNN encoder with window length six achieved the highest classification accuracy for DSAN, but failed to train properly for other window lengths. CoDATS and DANN were able to train properly for all window lengths, achieving classification accuracies between 40.90% to 60.26% and 42.90% to 61.67%, respectively. However, no statement can be made about the influence of window length on classification accuracy. Regarding the binary classification task, the CNN encoder outperforms the ResNet18 and TCN encoders. Depending on the window length, CDAN (up to 73.46%), HoMM_{p=3} (up to 74.94%), AdvSKM (up to 75.60%), and MMDA (up to 73.94%) achieves higher classification accuracies than other methods. A trend of higher classification accuracies for higher window lengths is evident for ResNet18 and TCN encoders.

Evaluation of DA & Encoder Networks on the Sinusoidal Datasets. Our study presents the evaluation results for all DA methods on the sinusoidal dataset illustrated in Figure 2.4, Figure 2.5, and Figure 2.6. Unlike previous datasets, where the CNN encoder had a superior performance over the ResNet18 and TCN encoders, on the sinusoidal datasets, the TCN encoder demonstrates robustness for all DA methods, while ResNet18 achieves low classification accuracies. It is important to note that KL and JSD are not considered proper DA methods, as indicated in Figure 2.5a. The CNN encoder is only applicable for DANN, Deep CORAL, DIRT-T, DDC, DAN, AdvSKM, MSE, and Sinkhorn, but limited to low noise parameters. The evaluation results reveal that MMDA (refer to Figure 2.4a), Deep CORAL and DANN (refer to Figure 2.4c), HoMM_{p=3} (refer to Figure 2.4d), and DDC and AdvSKM (refer to Figure 2.4e) demonstrate the highest robustness to noise. It is worth noting that although the F1-score is lower than the accuracy for DIRT-T, it remains the same for the other methods. The classification accuracy remains consistently high (up to 100%) for low noise parameters and decreases for higher noise parameters ($b > 0.7$). Based on the results, AdaTime concludes that joint distribution methods, including DIRT-T, MMDA, and DSAN, outperform marginal distribution methods, such as MMDA, Deep CORAL, and HoMM. The results obtained from sinusoidal datasets do not align with this statement. Jeffreys and Stein CORAL (see Figure 2.5b) exhibit similar performance as Deep CORAL (see Figure 2.4c). Although the performance differences between MMD (see Figure 2.6a) and MMCD (see Figure 2.6b) are marginal, the kernelized versions of MMD and MMCD show better results than the linear versions. According to the findings of the optimal transport method proposed by Ott et al. (2022a), Sinkhorn transport with L_pL_1 or L_1L_2 regularization outperforms the best DA method, which is MMDA. For high noise parameters ($b > 1.6$), MMDA's classification accuracy drops below 80%. In contrast, Sinkhorn transport with kMMD as a similarity comparison method consistently achieves classification accuracy above 80%. In our previous work (Ott et al., 2022a), we demonstrated that kMMD and Cosine similarity outperform CORAL, which in turn yields higher accuracies than HoMM of order three and Pearson correlation. As a result, the choice of encoder network and its hyperparameters significantly affects the final classification accuracy, and each DA method exhibits varying degrees of robustness

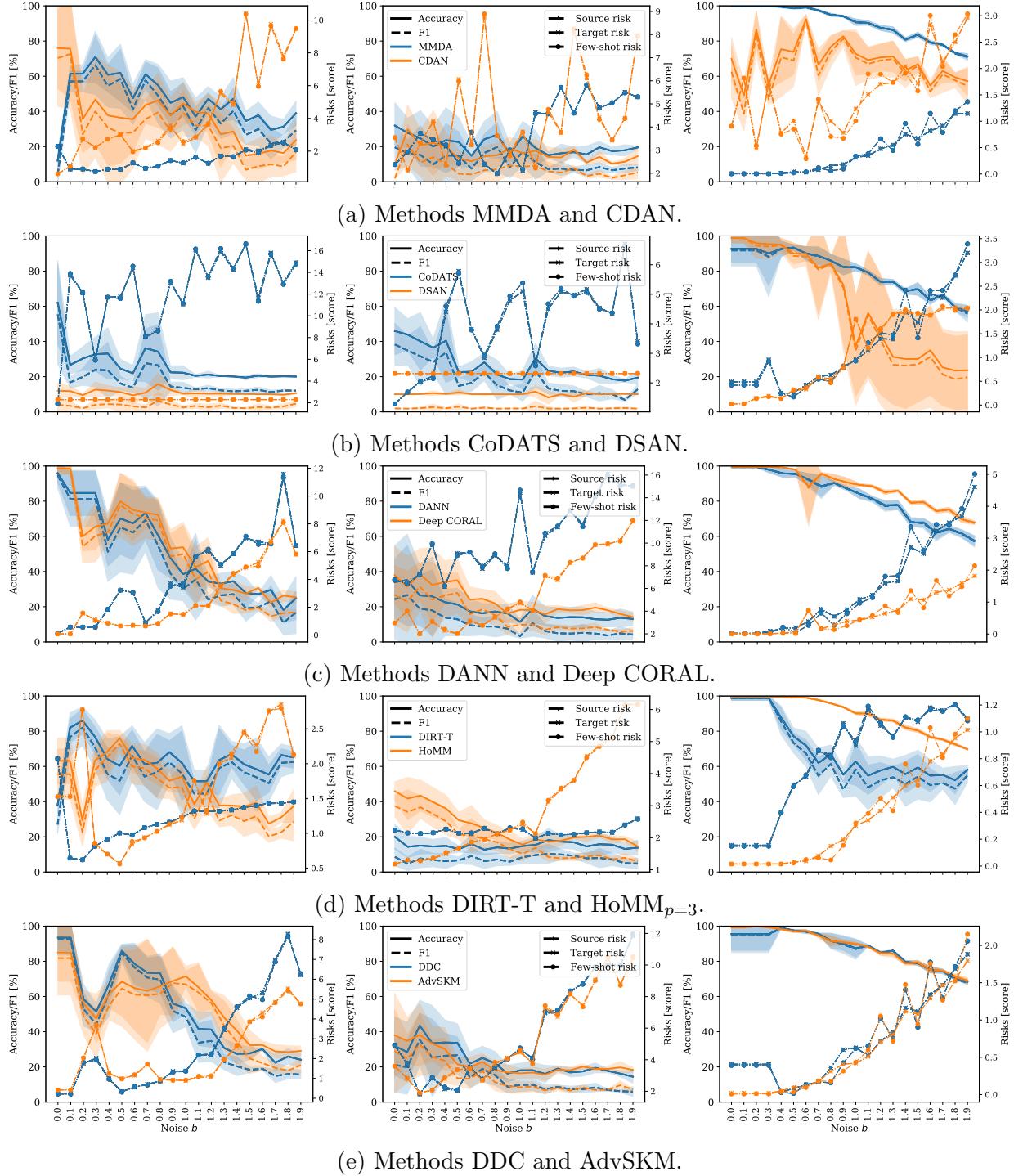


Figure 2.4: DA results for the sinusoidal datasets with noise parameter b . Left: CNN. Middle: ResNet18. Right: TCN. Results averaged over five runs.

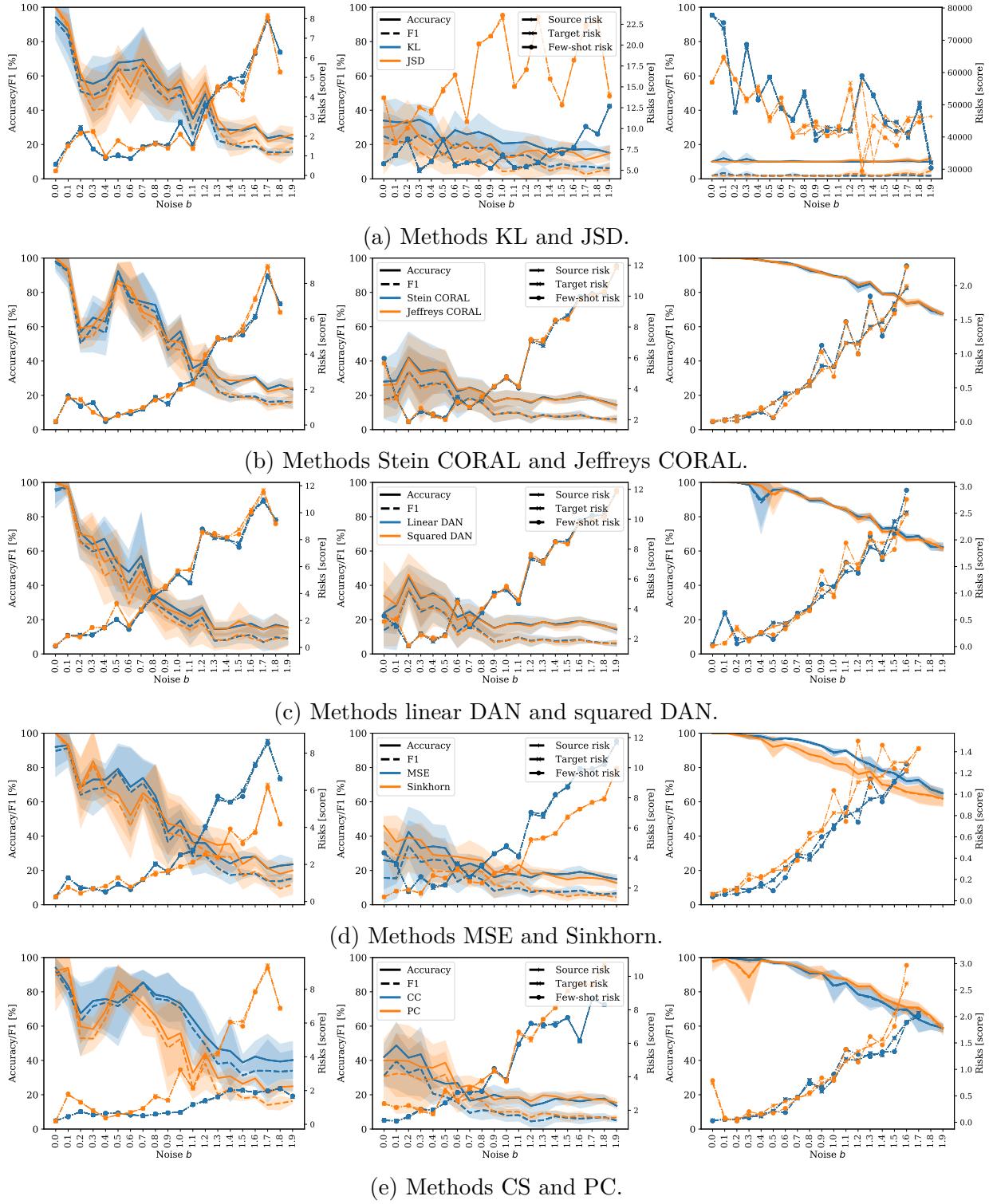


Figure 2.5: Figure 2.4 continued.

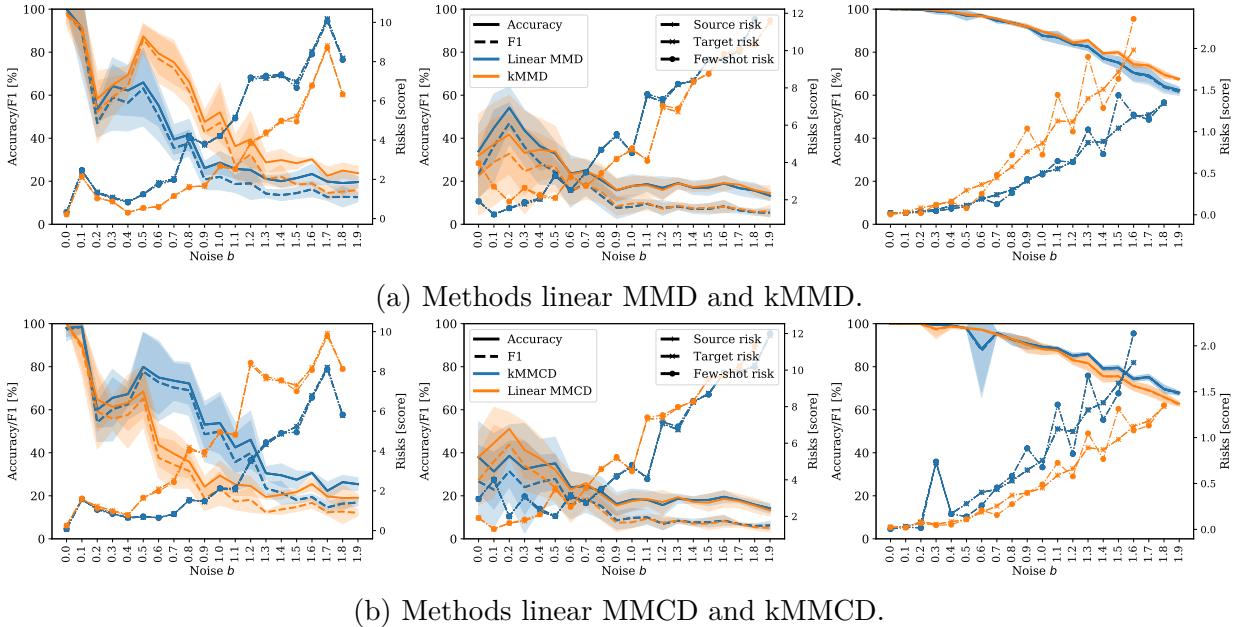
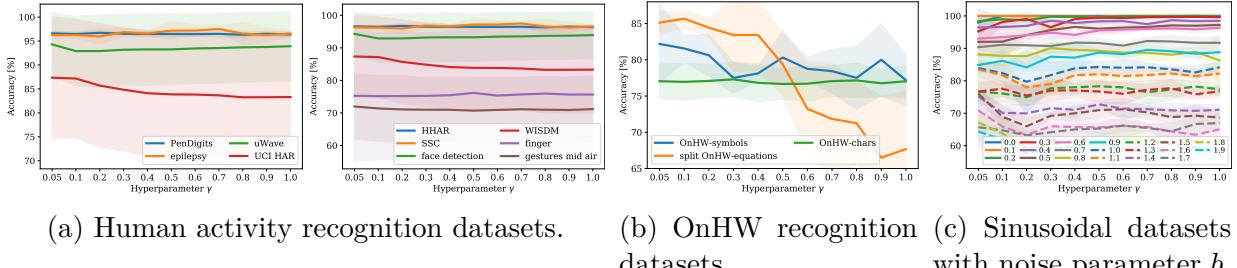


Figure 2.6: Figure 2.5 continued.

Figure 2.7: Hyperparameter search for the parameter $\gamma \in [0.05, 0.1, 0.2, 0.3, \dots, 1.0]$ of the Sinkhorn method. CNN encoder network for a) and b), and TCN encoder network for c). Mean accuracy and standard deviation (in %) given over five runs.

to noise. Therefore, a careful selection of the appropriate DA method for each application is critical to achieving a robust classifier.

Sinkhorn Hyperparameter Search. The Sinkhorn method's performance is dependent on the hyperparameter γ . We conducted a search to train Sinkhorn using $\gamma \in [0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]$ for all human activity, OnHW, and sinusoidal datasets (refer to Figure 2.7). Results indicate that $\gamma = 0.1$ achieves the highest accuracies for the human activity and OnHW datasets, while $\gamma = 0.05$ is inconsistent, resulting in higher accuracies for OnHW-symbols and WISDM datasets, but lower accuracies for the finger dataset. Furthermore, for specific noise parameters (i.e., $b = 0.2$, $b = 0.4$, $b = 0.5$, and $b = 0.6$) for the sinusoidal dataset, a higher γ improves classification accuracy. Specifi-

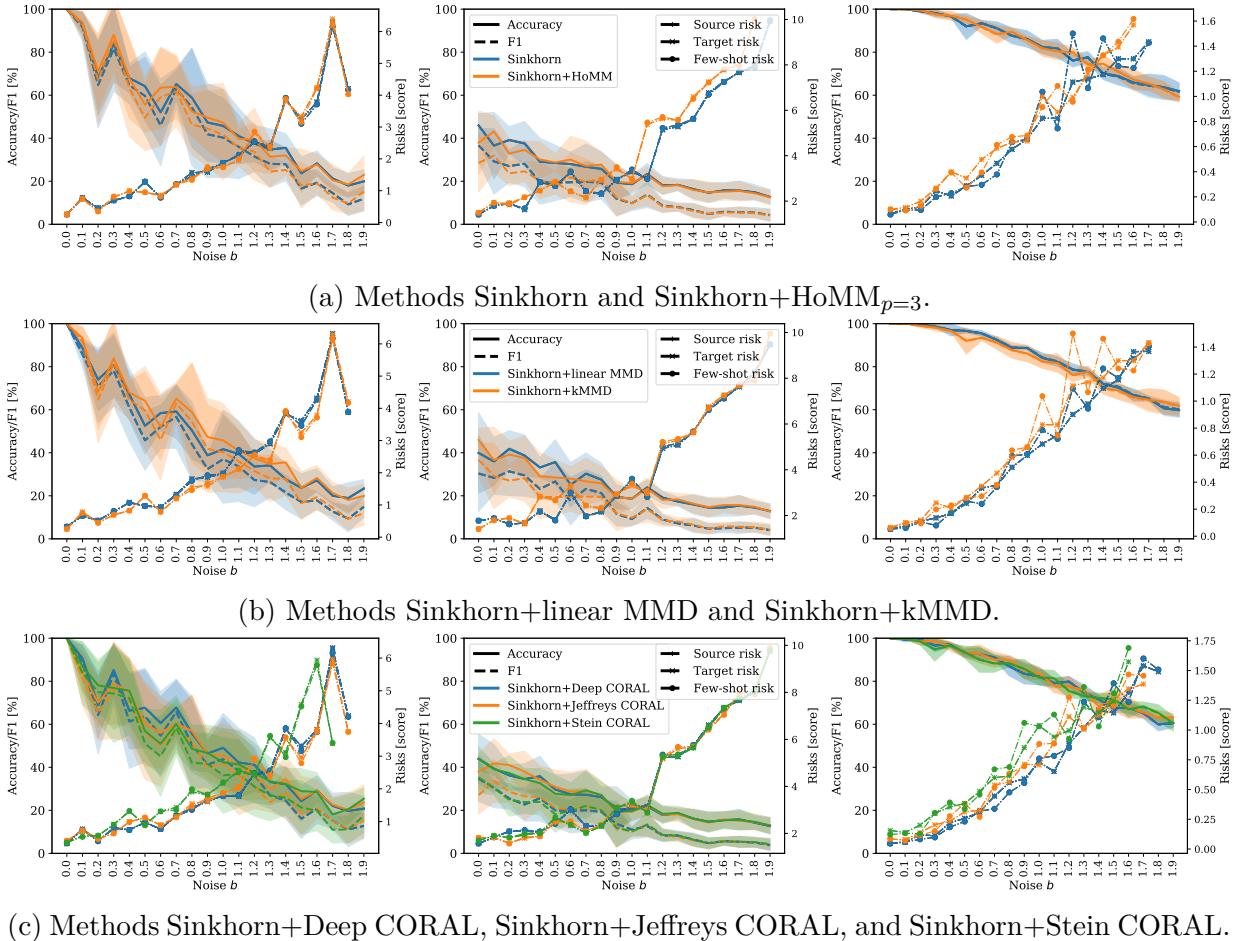


Figure 2.8: Figure 2.6 continued.

cally, for $\gamma = 0.05$, the accuracy decreases for low noise parameters ($b < 1.0$), but increases for high noise parameters ($b \geq 1.0$). To maintain consistency in training, we choose $\gamma = 0.1$ for follow-up trainings. The hyperparameter searches for CNN, ResNet18, and TCN encoder networks are similar concerning the parameter γ . For Sinkhorn, we employ 100 iterations, use MSE loss as the ground metric, and apply a stopping criterion of 0.1.

Combination of Sinkhorn & Alignment Functions. Feydy et al. (2019) demonstrated the advantageous geometric property of the Sinkhorn method and proved that it interpolates between OT and MMD. Thus, we aim to combine the strengths of both techniques by evaluating the benefits of feature alignment by representation learning (i.e., linear MMD, kMMD, Deep CORAL, Jeffreys CORAL, Stein CORAL, and HoMM_{p=3}) with the geometrical properties of Sinkhorn. We present results for ten human activity recognition datasets in Table 2.16 and Table 2.17, which can be compared with the results in Table 2.3 to Table 2.12. The Sinkhorn-only method with a parameter value of $\gamma = 0.1$ serves as the baseline for comparison. We assign equal weight to Sinkhorn and the align-

Table 2.16: DA results of the combination of Sinkhorn with alignment loss functions for the human activity recognition datasets. Mean and standard deviation in % over five runs. **Bold** are best results. Underlined are Sinkhorn-only improvements.

Datasets	Method	CNN		ResNet18		TCN	
		Accuracy	F1	Accuracy	F1	Accuracy	F1
HHAR	Sinkhorn ($\gamma = 0.1$)	75.15 \pm 19.5	73.97 \pm 20.7	67.98 \pm 20.9	67.18 \pm 22.5	70.04 \pm 19.1	68.11 \pm 21.4
	Sinkhorn+linear MMD	<u>75.17</u> \pm 19.4	73.83 \pm 20.6	67.82 \pm 21.3	67.11 \pm 22.8	70.49 \pm 19.7	<u>68.93</u> \pm 21.5
	Sinkhorn+kMMD	75.15 \pm 19.5	<u>73.97</u> \pm 20.7	67.98 \pm 20.9	67.18 \pm 22.5	70.04 \pm 19.1	68.11 \pm 21.4
	Sinkhorn+Deep CORAL	<u>75.19</u> \pm 19.3	73.84 \pm 20.5	66.88 \pm 21.7	66.09 \pm 23.3	70.81 \pm 19.4	69.04 \pm 21.5
	Sinkhorn+Jeffreys CORAL	75.40 \pm 19.8	74.10 \pm 21.0	67.29 \pm 21.8	66.49 \pm 23.3	70.76 \pm 18.6	<u>69.08</u> \pm 20.7
	Sinkhorn+Stein CORAL	74.69 \pm 19.4	73.43 \pm 20.5	68.45 \pm 21.3	67.62 \pm 22.8	71.08 \pm 19.4	69.34 \pm 21.6
	Sinkhorn+HoMM _{p=3}	75.14 \pm 19.1	73.90 \pm 20.3	67.36 \pm 21.8	66.49 \pm 23.4	70.68 \pm 19.1	68.83 \pm 21.4
UCI HAR	Sinkhorn ($\gamma = 0.1$)	87.17 \pm 12.4	86.55 \pm 12.9	81.62 \pm 12.7	78.96 \pm 14.0	80.88 \pm 12.9	79.05 \pm 13.5
	Sinkhorn+linear MMD	86.96 \pm 12.8	86.36 \pm 13.4	81.57 \pm 12.8	<u>79.00</u> \pm 14.1	81.27 \pm 12.6	79.58 \pm 13.3
	Sinkhorn+kMMD	87.17 \pm 12.4	86.55 \pm 12.9	81.62 \pm 12.7	78.96 \pm 14.0	80.88 \pm 12.9	79.05 \pm 13.5
	Sinkhorn+Deep CORAL	<u>87.25</u> \pm 12.4	<u>86.66</u> \pm 12.9	<u>81.67</u> \pm 12.7	78.95 \pm 14.1	<u>81.03</u> \pm 12.4	<u>79.21</u> \pm 13.1
	Sinkhorn+Jeffreys CORAL	87.04 \pm 12.5	86.45 \pm 13.1	81.49 \pm 13.0	78.83 \pm 14.3	<u>80.96</u> \pm 12.6	79.04 \pm 13.3
	Sinkhorn+Stein CORAL	87.12 \pm 12.5	86.48 \pm 13.1	81.50 \pm 12.9	78.95 \pm 14.1	<u>81.04</u> \pm 12.7	<u>79.22</u> \pm 13.3
	Sinkhorn+HoMM _{p=3}	87.30 \pm 12.4	86.68 \pm 13.0	82.95 \pm 13.0	80.87 \pm 14.1	81.00 \pm 12.6	<u>79.18</u> \pm 13.2
WISDM	Sinkhorn ($\gamma = 0.1$)	59.40 \pm 22.4	41.15 \pm 22.4	59.18 \pm 20.3	40.28 \pm 20.8	55.66 \pm 18.2	35.20 \pm 17.4
	Sinkhorn+linear MMD	59.28 \pm 22.1	40.99 \pm 22.3	58.75 \pm 20.9	39.98 \pm 21.0	55.55 \pm 18.6	35.02 \pm 17.6
	Sinkhorn+kMMD	59.40 \pm 22.4	41.15 \pm 22.4	59.18 \pm 20.3	40.28 \pm 20.8	55.66 \pm 18.2	35.20 \pm 17.4
	Sinkhorn+Deep CORAL	59.09 \pm 22.4	40.91 \pm 22.3	<u>59.22</u> \pm 20.5	<u>40.31</u> \pm 20.6	55.89 \pm 18.4	35.59 \pm 17.8
	Sinkhorn+Jeffreys CORAL	<u>59.51</u> \pm 22.5	<u>41.20</u> \pm 22.5	59.17 \pm 20.4	<u>40.40</u> \pm 20.8	55.59 \pm 18.3	35.34 \pm 17.8
	Sinkhorn+Stein CORAL	59.24 \pm 22.4	41.11 \pm 22.5	59.06 \pm 20.4	40.19 \pm 20.7	55.49 \pm 18.2	35.19 \pm 17.5
	Sinkhorn+HoMM _{p=3}	59.63 \pm 22.4	41.31 \pm 22.6	59.46 \pm 20.3	40.46 \pm 20.8	55.42 \pm 18.7	35.17 \pm 17.8
SSC EEG	Sinkhorn ($\gamma = 0.1$)	71.37 \pm 10.1	59.97 \pm 10.8	51.31 \pm 8.9	37.15 \pm 8.7	44.87 \pm 6.2	28.48 \pm 6.3
	Sinkhorn+linear MMD	71.57 \pm 10.0	60.17 \pm 10.8	51.25 \pm 8.8	37.13 \pm 8.6	44.49 \pm 6.3	28.15 \pm 6.2
	Sinkhorn+kMMD	71.37 \pm 10.1	59.97 \pm 10.8	51.31 \pm 8.9	37.15 \pm 8.7	44.87 \pm 6.2	28.48 \pm 6.3
	Sinkhorn+Deep CORAL	71.28 \pm 10.0	59.83 \pm 10.8	51.20 \pm 8.9	37.13 \pm 8.6	44.79 \pm 6.2	28.46 \pm 6.2
	Sinkhorn+Jeffreys CORAL	71.19 \pm 10.1	59.79 \pm 11.0	51.19 \pm 8.8	37.02 \pm 8.7	44.86 \pm 6.2	28.48 \pm 6.3
	Sinkhorn+Stein CORAL	71.60 \pm 10.2	<u>60.13</u> \pm 11.1	51.16 \pm 8.9	36.98 \pm 8.8	44.95 \pm 6.2	28.54 \pm 6.3
	Sinkhorn+HoMM _{p=3}	71.37 \pm 10.1	59.90 \pm 11.0	51.29 \pm 8.7	37.19 \pm 8.5	42.28 \pm 6.6	22.80 \pm 6.1
uWave	Sinkhorn ($\gamma = 0.1$)	92.89 \pm 7.3	92.41 \pm 8.2	58.01 \pm 10.0	54.21 \pm 11.3	84.08 \pm 13.2	83.01 \pm 14.0
	Sinkhorn+linear MMD	93.03 \pm 7.4	92.60 \pm 8.3	<u>58.05</u> \pm 10.4	54.46 \pm 11.8	85.19 \pm 11.7	84.20 \pm 12.3
	Sinkhorn+kMMD	92.89 \pm 7.3	92.41 \pm 8.2	58.01 \pm 10.0	54.21 \pm 11.3	84.08 \pm 13.2	83.01 \pm 14.0
	Sinkhorn+Deep CORAL	<u>93.13</u> \pm 7.1	<u>92.65</u> \pm 8.3	57.10 \pm 10.9	53.34 \pm 12.1	83.79 \pm 13.1	<u>83.05</u> \pm 13.6
	Sinkhorn+Jeffreys CORAL	92.98 \pm 7.1	<u>92.53</u> \pm 8.1	57.33 \pm 10.4	53.68 \pm 11.6	83.55 \pm 13.6	82.50 \pm 14.2
	Sinkhorn+Stein CORAL	93.10 \pm 7.3	<u>92.67</u> \pm 8.3	57.60 \pm 10.2	54.03 \pm 11.3	84.23 \pm 13.3	83.21 \pm 13.9
	Sinkhorn+HoMM _{p=3}	93.32 \pm 7.1	92.86 \pm 8.1	58.30 \pm 10.8	54.52 \pm 12.2	84.78 \pm 12.4	83.68 \pm 13.3
Finger	Sinkhorn ($\gamma = 0.1$)	50.35 \pm 4.9	46.11 \pm 4.9	48.33 \pm 4.2	42.32 \pm 6.1	55.07 \pm 1.1	35.88 \pm 2.2
	Sinkhorn+linear MMD	50.44 \pm 4.9	46.23 \pm 5.0	48.17 \pm 4.2	42.17 \pm 6.3	55.19 \pm 0.4	35.67 \pm 1.1
	Sinkhorn+kMMD	50.35 \pm 4.9	46.11 \pm 4.9	48.33 \pm 4.2	42.32 \pm 6.1	55.07 \pm 1.1	35.88 \pm 2.2
	Sinkhorn+Deep CORAL	50.32 \pm 5.0	46.06 \pm 4.9	47.46 \pm 4.2	42.50 \pm 6.2	55.11 \pm 0.9	35.86 \pm 2.1
	Sinkhorn+Jeffreys CORAL	50.37 \pm 5.0	46.13 \pm 5.0	48.40 \pm 4.2	42.47 \pm 6.2	55.09 \pm 1.0	35.91 \pm 2.3
	Sinkhorn+Stein CORAL	50.37 \pm 4.9	46.13 \pm 4.9	48.43 \pm 4.3	42.48 \pm 6.3	55.11 \pm 0.9	35.88 \pm 2.2
	Sinkhorn+HoMM _{p=3}	50.33 \pm 4.9	46.10 \pm 4.9	48.29 \pm 4.2	42.18 \pm 6.1	54.90 \pm 1.5	37.84 \pm 5.7
Ges-tures mid air	Sinkhorn ($\gamma = 0.1$)	41.18 \pm 14.3	35.22 \pm 15.3	30.19 \pm 11.8	23.25 \pm 11.9	28.66 \pm 12.1	22.44 \pm 12.3
	Sinkhorn+linear MMD	41.41 \pm 14.3	35.44 \pm 15.2	29.94 \pm 12.2	22.97 \pm 12.2	27.91 \pm 11.9	21.17 \pm 12.2
	Sinkhorn+kMMD	41.18 \pm 14.3	35.22 \pm 15.3	30.19 \pm 11.8	23.25 \pm 11.9	28.66 \pm 12.1	22.44 \pm 12.3
	Sinkhorn+Deep CORAL	40.96 \pm 14.5	<u>35.31</u> \pm 15.4	30.15 \pm 12.2	23.22 \pm 12.2	28.57 \pm 12.2	21.88 \pm 12.5
	Sinkhorn+Jeffreys CORAL	41.09 \pm 14.3	35.00 \pm 15.0	30.18 \pm 11.3	23.13 \pm 11.3	28.37 \pm 12.0	21.70 \pm 12.2
	Sinkhorn+Stein CORAL	41.38 \pm 14.4	<u>35.35</u> \pm 15.3	30.02 \pm 12.0	23.14 \pm 11.9	28.89 \pm 12.6	22.31 \pm 12.7
	Sinkhorn+HoMM _{p=3}	41.00 \pm 14.3	35.04 \pm 15.3	29.22 \pm 12.0	22.17 \pm 11.9	27.62 \pm 12.2	21.02 \pm 12.3

ment loss functions (weighting of 1). Depending on the dataset, the alignment function can improve the Sinkhorn results further. For instance, Jeffreys CORAL on the HHAR dataset, HoMM_{p=3} on the UCI HAR, WISDM, and uWave datasets, and Stein CORAL

Table 2.17: Table 2.16 continued.

Datasets	Method	CNN		ResNet18		TCN	
		Accuracy	F1	Accuracy	F1	Accuracy	F1
Epilepsy	Sinkhorn ($\gamma = 0.1$)	96.25 \pm 0.8	96.28 \pm 0.8	90.31 \pm 4.6	89.72 \pm 5.2	64.69 \pm 4.4	63.84 \pm 4.5
	Sinkhorn+linear MMD	96.25 \pm 0.8	96.28 \pm 0.8	90.94 \pm 4.4	90.60 \pm 4.6	65.62 \pm 1.4	63.95 \pm 1.1
	Sinkhorn+kMMD	96.25 \pm 0.8	96.28 \pm 0.8	90.31 \pm 4.6	89.72 \pm 5.2	64.69 \pm 4.4	63.84 \pm 4.5
	Sinkhorn+Deep CORAL	96.88 \pm 2.2	<u>96.88</u> \pm 2.3	94.69 \pm 2.1	94.53 \pm 2.3	64.69 \pm 1.9	63.06 \pm 2.5
	Sinkhorn+Jeffreys CORAL	96.25 \pm 1.6	96.20 \pm 1.6	91.88 \pm 6.2	91.43 \pm 7.0	63.44 \pm 4.1	62.33 \pm 4.1
	Sinkhorn+Stein CORAL	96.25 \pm 0.8	96.28 \pm 0.8	90.94 \pm 6.2	89.83 \pm 7.2	63.44 \pm 3.5	61.59 \pm 3.6
	Sinkhorn+HoMM _{p=3}	96.88 \pm 1.0	96.93 \pm 1.0	93.12 \pm 6.4	92.27 \pm 8.1	61.88 \pm 3.2	60.97 \pm 3.7
Face detection	Sinkhorn ($\gamma = 0.1$)	61.16 \pm 1.9	59.61 \pm 2.8	52.85 \pm 1.3	50.27 \pm 3.0	59.85 \pm 7.2	53.13 \pm 15.4
	Sinkhorn+linear MMD	<u>62.12</u> \pm 1.0	<u>60.76</u> \pm 1.8	52.09 \pm 0.7	48.48 \pm 2.7	55.64 \pm 7.3	45.68 \pm 15.3
	Sinkhorn+kMMD	61.16 \pm 1.9	59.61 \pm 2.8	52.85 \pm 1.3	50.27 \pm 3.0	59.85 \pm 7.2	53.13 \pm 15.4
	Sinkhorn+Deep CORAL	<u>62.07</u> \pm 1.2	60.80 \pm 1.9	52.70 \pm 1.2	51.21 \pm 2.0	64.63 \pm 1.7	64.47 \pm 1.7
	Sinkhorn+Jeffreys CORAL	60.56 \pm 2.0	59.35 \pm 2.0	53.38 \pm 0.9	52.87 \pm 1.0	<u>62.41</u> \pm 6.7	<u>59.10</u> \pm 13.1
	Sinkhorn+Stein CORAL	61.19 \pm 0.9	59.48 \pm 2.2	50.82 \pm 0.8	46.01 \pm 5.7	<u>62.84</u> \pm 6.9	<u>59.55</u> \pm 13.3
	Sinkhorn+HoMM _{p=3}	62.56 \pm 1.8	61.41 \pm 2.8	52.52 \pm 1.9	47.46 \pm 5.8	<u>63.40</u> \pm 7.1	<u>60.15</u> \pm 13.6
Pen-Digits	Sinkhorn ($\gamma = 0.1$)	96.46 \pm 0.3	96.43 \pm 0.3	76.29 \pm 0.9	76.08 \pm 1.1	72.07 \pm 6.4	67.48 \pm 8.1
	Sinkhorn+linear MMD	<u>96.51</u> \pm 0.2	<u>96.47</u> \pm 0.2	76.45 \pm 0.7	<u>76.17</u> \pm 0.9	<u>73.45</u> \pm 14.7	69.84 \pm 16.3
	Sinkhorn+kMMD	96.46 \pm 0.3	96.43 \pm 0.3	76.29 \pm 0.9	76.08 \pm 1.1	72.07 \pm 6.4	67.48 \pm 8.1
	Sinkhorn+Deep CORAL	96.63 \pm 0.4	96.60 \pm 0.4	76.18 \pm 1.0	76.05 \pm 1.1	69.77 \pm 9.9	65.36 \pm 11.2
	Sinkhorn+Jeffreys CORAL	96.63 \pm 0.3	<u>96.58</u> \pm 0.3	76.25 \pm 1.0	76.01 \pm 1.0	70.08 \pm 7.0	64.84 \pm 8.3
	Sinkhorn+Stein CORAL	96.42 \pm 0.4	96.38 \pm 0.5	<u>76.38</u> \pm 0.9	76.20 \pm 1.1	73.55 \pm 5.0	<u>69.35</u> \pm 6.2
	Sinkhorn+HoMM _{p=3}	96.25 \pm 0.5	96.20 \pm 0.5	<u>76.37</u> \pm 0.9	<u>76.09</u> \pm 0.9	52.28 \pm 4.3	47.10 \pm 4.7

Table 2.18: DA results of the combination of Sinkhorn with alignment loss functions for the OnHW recognition datasets (L \rightarrow R). Mean and standard deviation in % over five runs. **Bold** are best results. Underline are Sinkhorn-only improvements.

Datasets	Method	CNN		ResNet18		TCN	
		Accuracy	F1	Accuracy	F1	Accuracy	F1
OnHW-symbols	Sinkhorn ($\gamma = 0.1$)	81.56 \pm 2.0	82.77 \pm 1.8	45.31 \pm 5.5	42.36 \pm 5.4	6.25 \pm 1.1	1.17 \pm 1.0
	Sinkhorn+linear MMD	79.69 \pm 3.1	81.29 \pm 3.4	42.81 \pm 11.5	40.04 \pm 12.5	6.25 \pm 1.1	<u>1.19</u> \pm 0.6
	Sinkhorn+kMMD	81.56 \pm 2.0	82.77 \pm 1.8	54.31 \pm 5.5	42.36 \pm 5.4	6.25 \pm 1.1	1.17 \pm 1.0
	Sinkhorn+Deep CORAL	80.94 \pm 3.6	82.77 \pm 3.5	38.12 \pm 9.3	34.67 \pm 8.8	<u>7.50</u> \pm 3.4	1.03 \pm 0.3
	Sinkhorn+Jeffreys CORAL	80.31 \pm 2.6	81.57 \pm 3.3	38.75 \pm 5.7	35.04 \pm 7.3	<u>7.19</u> \pm 3.0	1.72 \pm 1.4
	Sinkhorn+Stein CORAL	77.50 \pm 3.8	78.47 \pm 3.8	38.12 \pm 13.5	34.28 \pm 17.0	6.25 \pm 0.0	<u>1.92</u> \pm 1.7
	Sinkhorn+HoMM _{p=3}	79.06 \pm 5.6	80.09 \pm 5.3	42.50 \pm 9.5	39.19 \pm 9.8	9.06 \pm 1.7	<u>4.51</u> \pm 1.7
Split OnHW-equations	Sinkhorn ($\gamma = 0.1$)	85.67 \pm 1.4	84.33 \pm 2.1	70.85 \pm 5.3	70.58 \pm 4.9	5.95 \pm 2.2	1.39 \pm 0.8
	Sinkhorn+linear MMD	84.44 \pm 5.3	83.56 \pm 5.7	68.80 \pm 5.3	67.82 \pm 5.2	<u>7.93</u> \pm 2.2	<u>1.80</u> \pm 1.9
	Sinkhorn+kMMD	85.67 \pm 1.4	84.33 \pm 2.1	70.85 \pm 5.3	70.58 \pm 4.9	5.95 \pm 2.2	1.39 \pm 0.8
	Sinkhorn+Deep CORAL	86.31 \pm 2.2	85.30 \pm 2.2	72.04 \pm 1.3	71.14 \pm 1.2	8.01 \pm 0.0	0.99 \pm 0.0
	Sinkhorn+Jeffreys CORAL	85.85 \pm 3.1	84.56 \pm 3.9	71.51 \pm 4.8	71.26 \pm 5.2	8.01 \pm 0.0	0.99 \pm 1.0
	Sinkhorn+Stein CORAL	85.99 \pm 1.3	84.79 \pm 1.3	72.28 \pm 2.2	71.68 \pm 2.4	8.17 \pm 2.1	<u>3.10</u> \pm 1.8
	Sinkhorn+HoMM _{p=3}	85.77 \pm 3.8	<u>84.86</u> \pm 4.3	69.15 \pm 2.3	68.20 \pm 2.8	8.61 \pm 1.3	1.36 \pm 0.8
OnHW-chars	Sinkhorn ($\gamma = 0.1$)	76.96 \pm 2.5	75.27 \pm 3.4	37.21 \pm 3.8	34.48 \pm 4.0	1.94 \pm 0.4	0.26 \pm 0.1
	Sinkhorn+linear MMD	76.80 \pm 2.1	75.02 \pm 3.5	<u>38.56</u> \pm 2.7	<u>35.94</u> \pm 3.3	1.76 \pm 0.4	<u>0.27</u> \pm 0.1
	Sinkhorn+kMMD	76.96 \pm 2.5	75.27 \pm 3.4	37.21 \pm 3.8	34.48 \pm 4.0	1.94 \pm 0.4	0.26 \pm 0.1
	Sinkhorn+Deep CORAL	77.19 \pm 2.4	75.20 \pm 3.8	36.64 \pm 5.2	34.12 \pm 4.9	0.69 \pm 0.2	0.04 \pm 0.0
	Sinkhorn+Jeffreys CORAL	77.97 \pm 1.6	77.00 \pm 1.3	<u>38.44</u> \pm 4.0	<u>35.58</u> \pm 4.3	0.69 \pm 0.2	0.04 \pm 0.0
	Sinkhorn+Stein CORAL	76.91 \pm 2.7	74.98 \pm 4.0	38.63 \pm 5.1	35.97 \pm 5.4	1.92 \pm 0.0	<u>0.29</u> \pm 0.0
	Sinkhorn+HoMM _{p=3}	76.91 \pm 3.1	74.85 \pm 4.4	<u>38.32</u> \pm 3.7	<u>35.70</u> \pm 3.7	2.13 \pm 0.2	<u>0.39</u> \pm 0.1

on the SSC dataset (for the CNN encoder network). Although HoMM_{p=3}-only achieves better results on the HHAR and UCI HAR datasets compared to the combination with Sinkhorn, HoMM_{p=3}-only fails on the WISDM dataset. Notably, the combination benefits

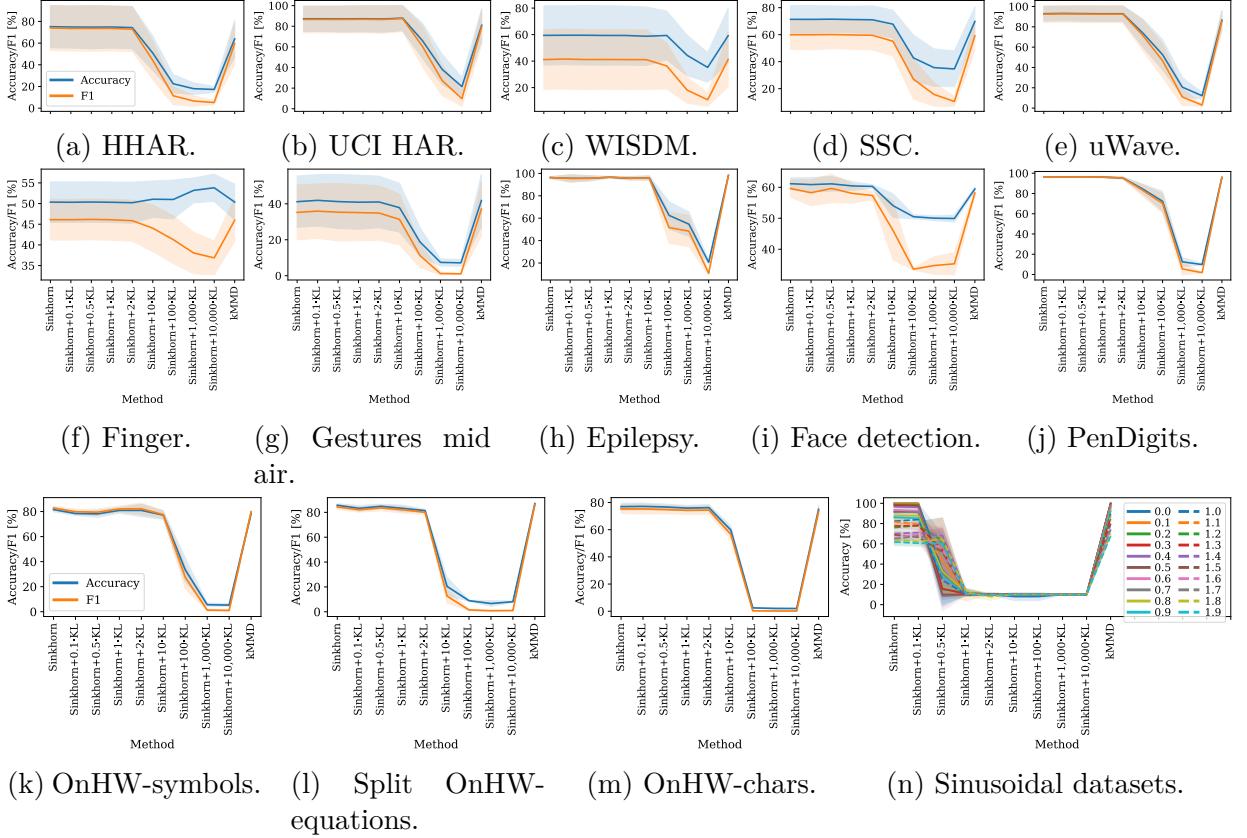


Figure 2.9: Combination of Sinkhorn with KL for the different weighting parameters $\epsilon \in [0.1, 0.5, 1, 2, 10, 100, 1,000, 10,000]$ for the 10 human activity, the OnHW, and the sinusoidal datasets. Results are averaged over five runs and all DA scenarios.

the SSC dataset, where Stein CORAL-only achieves 69.95% and Sinkhorn-only achieves 71.37% accuracy, but the combination of both methods results in an accuracy of 71.60%. A similar pattern is observed for the uWave dataset, where $\text{HoMM}_{p=3}$ -only results in 91.00% and Sinkhorn-only achieves 92.89% classification accuracy, Sinkhorn+ $\text{HoMM}_{p=3}$ improves the result (93.32%). While the combination of Sinkhorn and the alignment functions show marginal improvements on the finger, gestures mid air, and epilepsy datasets. However, we found that Sinkhorn combined with Deep CORAL significantly improved the results for the face detection dataset (64.63%) and the TCN encoder. Nonetheless, it was still unable to surpass the accuracy achieved by $\text{HoMM}_{p=3}$ -only. The results for the OnHW datasets are presented in Table 2.18 (the baseline results can be found in Table 2.13 to Table 2.15). For the split OnHW-equations dataset, Sinkhorn+Deep CORAL (86.31%) outperforms Sinkhorn-only (85.67%) and Deep CORAL-only (84.66%), but linear MMCD and linear DAN (both 88.37%) still yield the highest classification results. For OnHW-chars, Sinkhorn+Jeffreys CORAL (77.97%) is unable to surpass the $\text{HoMM}_{p=3}$ -only result (78.14%). The sinusoidal dataset results (see Figure 2.8) show that all methods perform similarly and do not outperform MMDA (see Figure 2.4a) and kMMCD (see Figure 2.6b).

As a result, it cannot be concluded that any combination of methods will be optimal for all datasets and encoder networks. The combination of methods and hyperparameters needs to be carefully chosen for each application. Figure 2.9 present the results of combining Sinkhorn+ ϵ ·KL to investigate the Sinkhorn regularization property discussed in Section 1.5.4, where S_ϵ approaches the MMD distance as $\epsilon \rightarrow \infty$. The results are consistent across all datasets for $\epsilon \in [0.1, 0.5, 1, 2]$. However, for larger values of ϵ such as $\epsilon \geq 0.5$, $\epsilon \geq 10$, or $\epsilon \geq 100$, the performance significantly decreases, although this varies depending on the dataset. It should be noted that the results for kMMD are similar to those obtained using Sinkhorn-only, and thus Sinkhorn+ ϵ ·KL does not converge to MMD as $\epsilon \rightarrow \infty$.

Evaluation of Risks. The selection of the appropriate model, method, and corresponding hyperparameters for unsupervised DA usually relies on the accuracy on the target domain, which violates the key assumption of unsupervised DA as the target domain labels are not available during training. To address this issue, AdaTime (Ragab et al., 2023) utilized the source risk (Ganin et al., 2016) and validation risk (You et al., 2019), which do not require any target domain labels. Additionally, the few-shot target risk employs a small number of labeled samples (here, five samples per class) from the target train domain. In our evaluation, we present the source risk, target risk, and few-shot target risk. The source risk refers to the cross-entropy loss on the test set from the source domain, while the target risk is computed using samples from the target domain as the validation set and is also measured by the cross-entropy loss. The hyperparameters selected using the target domain are the upper bound for the unsupervised DA method. The few-shot target risk uses only a few annotated samples per class of the target domain. However, in real-world scenarios, obtaining a large amount of labeled data from the target domain is often not possible. In such cases, a few labeled samples per class from the target domain can still be used to compute the few-shot target risk, which is calculated using the cross-entropy loss. Figure 2.10 presents the accuracy and the three risks on the human activity and sleep recognition datasets for all possible domain combinations, for instance, for the CNN encoder and HoMM_{p=3}. The average accuracy and F1-scores are shown using dashed horizontal lines. Please note that for readability, not every ID of the source and target domain is shown. The source risk remains consistently low for all domain combinations, but the target and few-shot target risks are found to be correlated with the classification accuracy and the F1-score, and they vary with the domain combination. For instance, in the HHAR dataset (see Figure 2.10a), the target and few-shot target risks drop below 0.2 for the domain IDs 1→3, 1→4, and 1→5 where the classification accuracy is high, and vice versa for the domain ID 1→2. Similar observations can be made for the uWave dataset (see Figure 2.10e), for instance, for the domain IDs 1→6 and 1→8. The results provide insights into the similarity of data between domains (i.e., participants). The difference between target and few-shot target risks is marginal for all datasets, and both risks are correlated. Therefore, the few-shot target risk is an appropriate measure for model and hyperparameter selection. Figure 2.11 depicts the risks on the OnHW datasets, and the pattern is similar to the previously mentioned datasets. For the GNSS-based interfer-

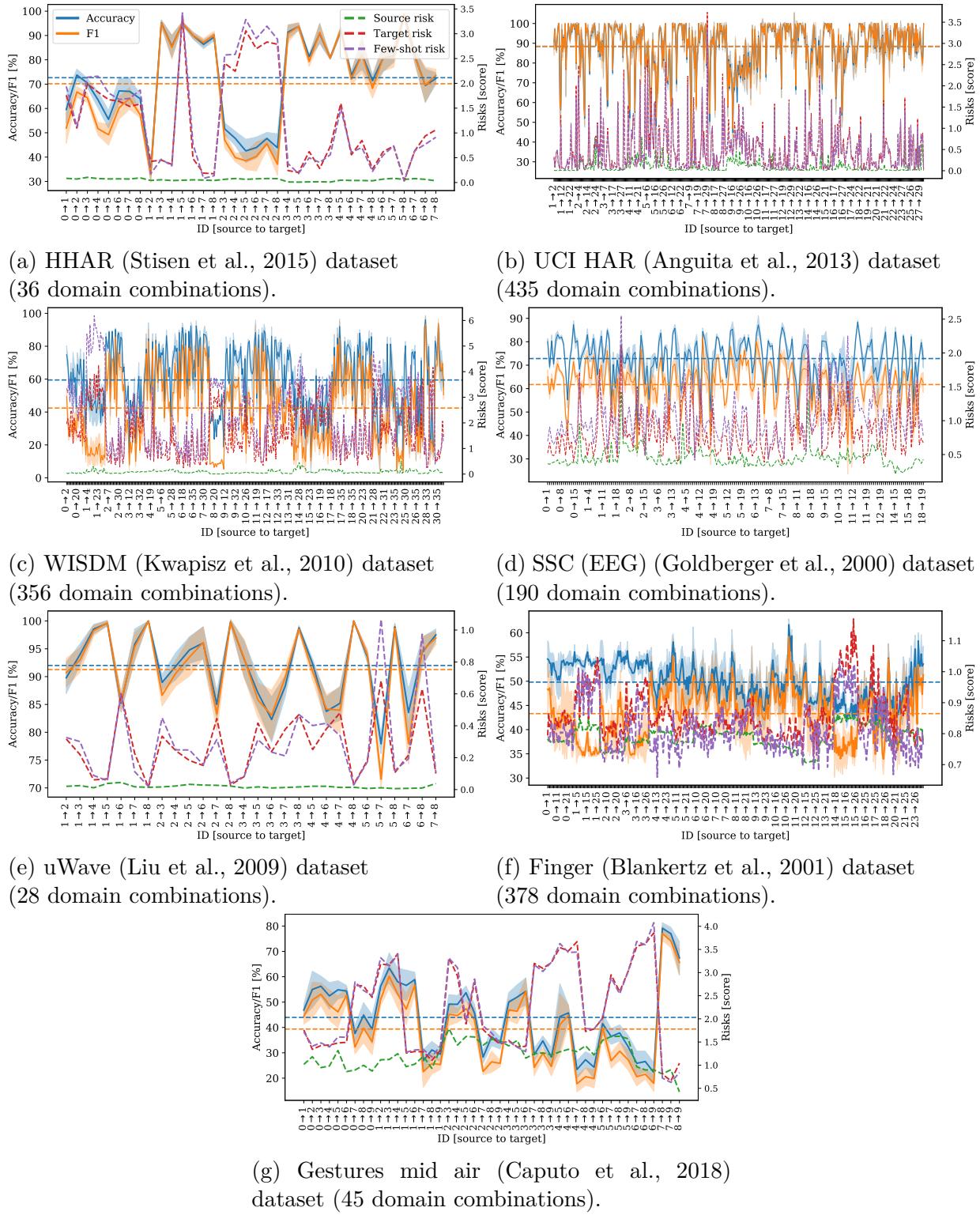


Figure 2.10: DA results for different source to target domains averaged over five runs.

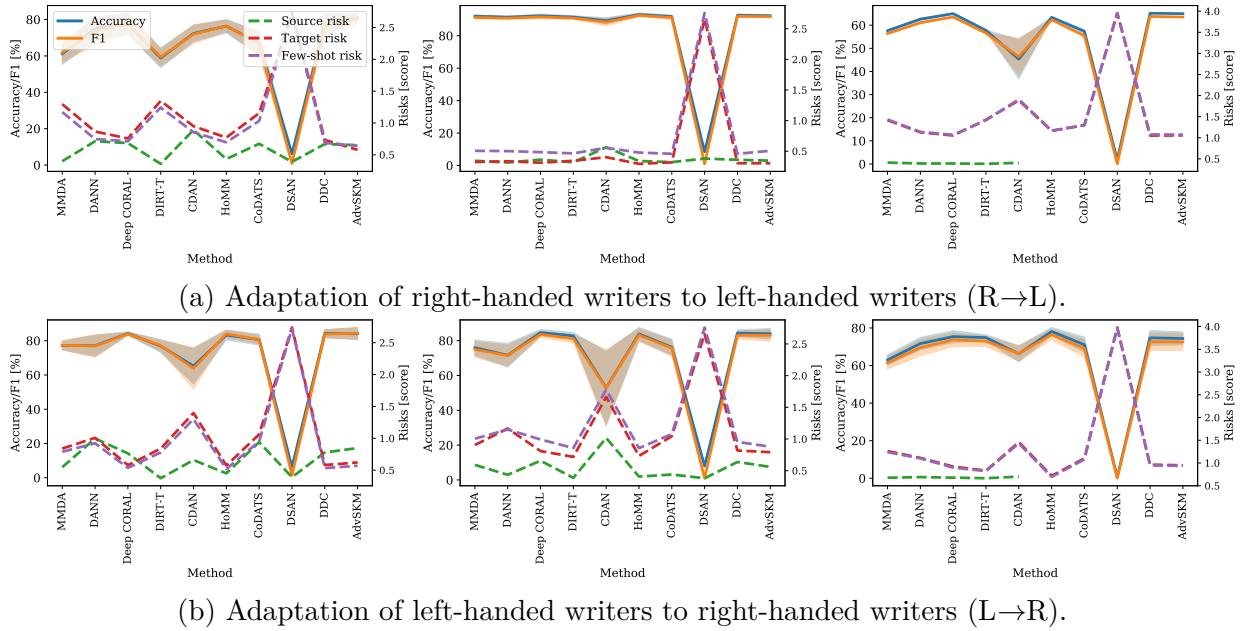


Figure 2.11: DA and risk results for the OnHW datasets and the CNN encoder for left-handed and right-handed writers. Left: OnHW-symbols. Middle: split OnHW-equations. Right: OnHW-chars. Results are averaged over five runs for the OnHW-symbols and split OnHW-equations datasets, and additionally averaged over all five cross-validation splits for the OnHW-chars datasets.

ence detection dataset (see Figure 2.3), the source risk is low (< 1.0), but the difference between target and few-shot target risks is higher for the binary classification task (see Figure 2.3d). This difference becomes more pronounced for the multi-class problem (see Figure 2.3a and Figure 2.3b). In the case of the CNN encoder, the source risk ranges from 1 to 2, while the target risk ranges from 2 to 4. However, the few-shot target risk increases significantly up to 12 on the GNSS dataset. Consequently, relying solely on the few-shot risk may result in erroneous model selections. For the sinusoidal datasets, the source and target risks are consistently similar, and are positively correlated with the classification accuracy, as demonstrated in Figure 2.4. Specifically, MMDA in Figure 2.4a and DANN and Deep CORAL in Figure 2.4b) exhibit this trend. As the noise parameters increase, the difference between the target and few-shot target risks also increases. Even though the target risk shows consistent growth, the few-shot target risk varies significantly. As a result, selecting the appropriate number of samples for the few-shot target risk is critical for model selection. In the case of the sinusoidal dataset, choosing few samples for the few-shot target risk can lead to inappropriate model selection.

Evaluation w.r.t. the Upper Bound. Figure 2.12 and Figure 2.13 provide a summary of the results obtained for the ten human activity recognition datasets, as well as the three OnHW datasets, and display the upper bound (refer to Table 2.2 for upper and lower

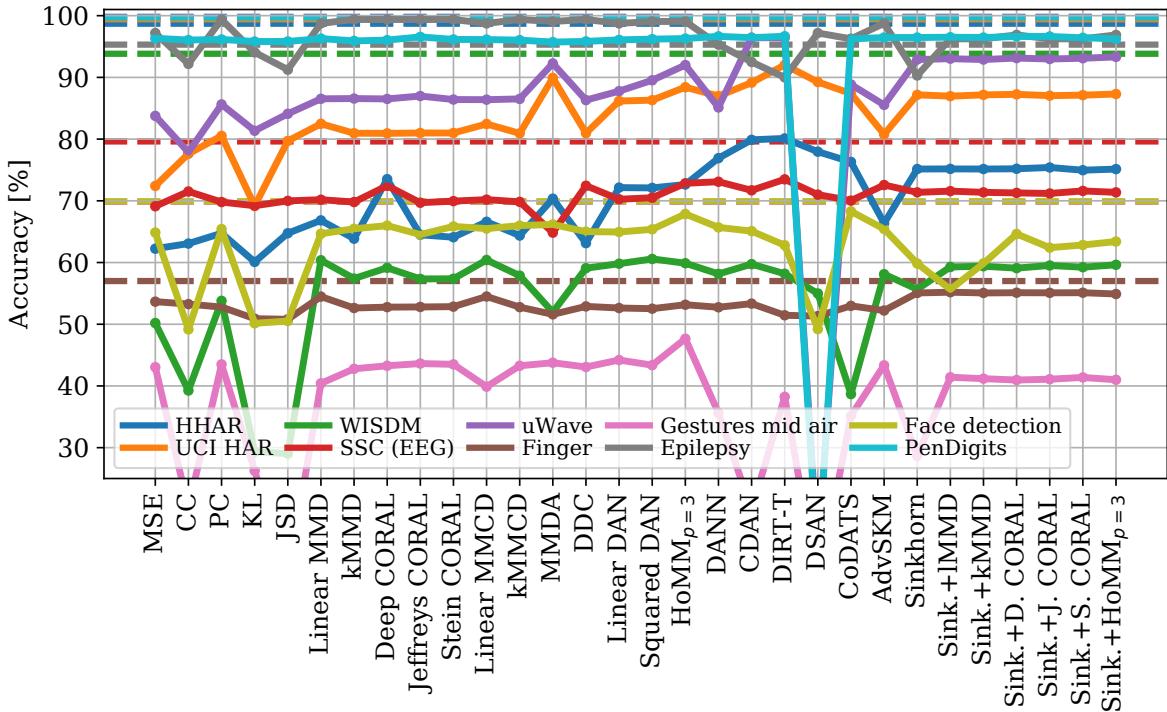


Figure 2.12: Comparison of all DA methods on the 10 human activity datasets with respect to the upper bound. Dashed lines are the upper bounds.

bounds). Out of the 10 datasets, the upper bound is attained for 5, specifically for the epilepsy (alignment methods), finger movements (Sinkhorn methods), and face detection ($\text{HoMM}_{p=3}$ and CoDATS) datasets. However, the results for the WISDM dataset are significantly distant from the upper bound. DA methods show higher effectiveness for the OnHW datasets, with the upper bound being achieved for all three datasets, reducing the domain shift between left-handed and right-handed writers. Linear MMD and linear MMCD attain the upper bound for the split OnHW-equations dataset, while Deep CORAL, DDC, $\text{HoMM}_{p=3}$, AdvSKM, and OT [kMMD] achieve results higher than the upper bound for the OnHW-symbols dataset, and $\text{HoMM}_{p=3}$ and the Sinkhorn combinations outperform the upper bound for OnHW-chars.

Training Times. The training times for all 30 methods and loss combinations on the sinusoidal datasets are compared (results are averaged over 100 trainings for each method). An overview of the training times (in s) is provided in Table 2.19, with small standard deviation observed (ranging from 0.4 to 4.2). The baseline training time of approximately 107 s is observed for models utilizing the standard loss functions MSE, CS, PC, KL, and JSD, as well as for the linear MMD method (107.6 s). The kernelized versions of MMD, namely kMMD (74.2 s) and DDC (74.2 s), exhibit significantly faster training times. Deep CORAL, a second-order method, trains at a similar speed (107.5 s), while Jeffreys CORAL (140.4 s) and Stein CORAL (150.6 s) require more computing time due to the utilization of

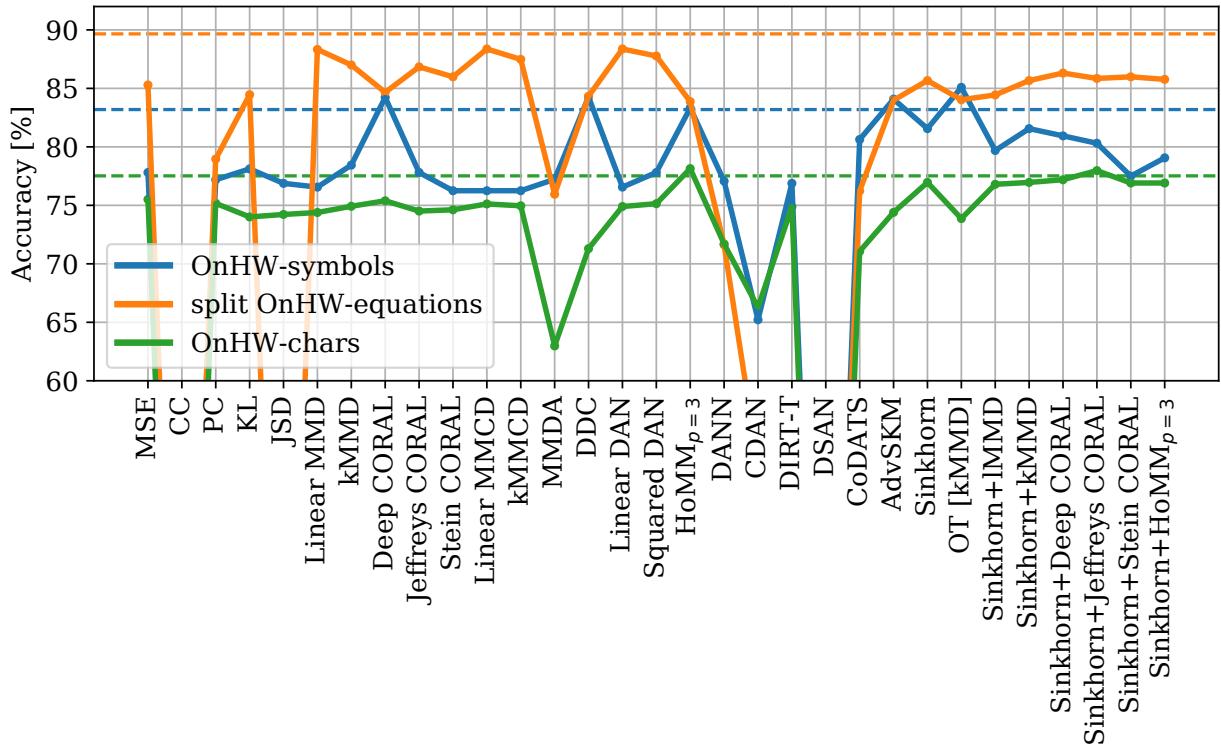


Figure 2.13: Comparison of all DA methods on the three online handwriting datasets with respect to the upper bound. Dashed lines are the upper bounds.

Table 2.19: Overview of training times (in s) exemplary on the sinusoidal dataset (TCN encoder) averaged over all 20 datasets with different noise parameters and five runs.

Method	Time (in s)	Method	Time (in s)	Method	Time (in s)
MSE	107.1	Linear MMCD	108.3	DSAN	125.2
CS	107.2	Kernelized MMCD	108.1	CoDATS	108.3
PC	107.2	MMDA	108.7	AdvSKM	75.9
KL	107.2	DDC	74.2	Sinkhorn	220.0
JSD	107.3	Linear DAN	109.3	Sinkhorn+linear MMD	218.0
Linear MMD	107.6	Squared DAN	335.6	Sinkhorn+kMMD	222.0
kMMD	74.2	HoMM $_{p=3}$	122.3	Sinkhorn+Deep CORAL	220.4
Deep CORAL	107.5	DANN	107.9	Sinkhorn+Jeffreys CORAL	322.2
Jeffreys CORAL	140.4	CDAN	109.5	Sinkhorn+Stein CORAL	328.7
Stein CORAL	150.6	DIRT-T	306.1	Sinkhorn+HoMM $_{p=3}$	220.8

the log determinant and matrix inverse. The linear DAN method has a fast training time (109.3 s), but this significantly increases for the squared DAN method (335.6 s). Consequently, squared DAN requires over 15 days for all DA scenarios on the large EEG dataset. The third-order domain alignment loss HoMM $_{p=3}$ further increases training times (122.3 s). While DIRT-T outperforms other methods on five of the ten human activity recognition

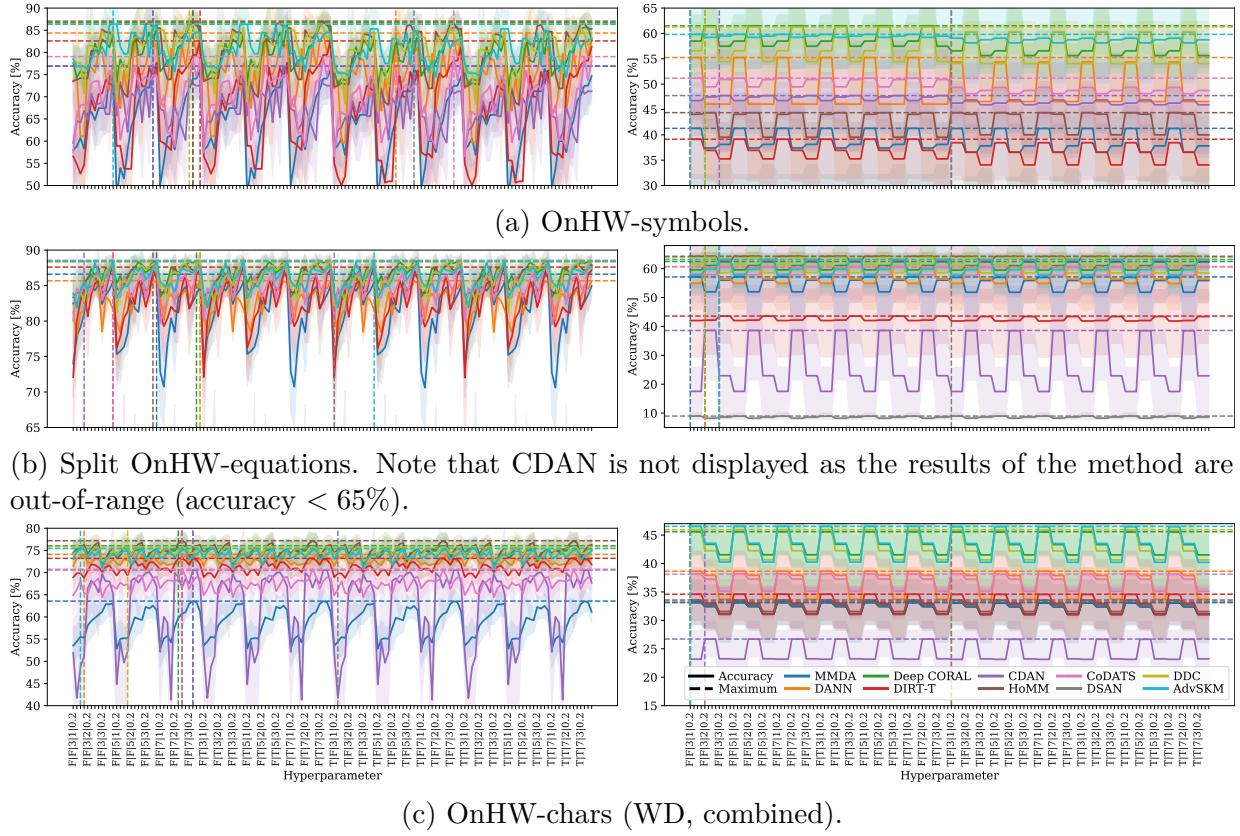


Figure 2.14: Hyperparameter search for the OnHW datasets (L→R) for ten DA methods. Left: CNN encoder. Right: ResNet18 encoder. Results are averaged over five runs. Hyperparameters are sorted the following: $drop\ last \in [False, True]$, $normalize \in [False, True]$, $kernel\ size \in [3, 5, 7]$, $stride \in [1, 2, 3]$, and $dropout\ rate \in [0.2, 0.3, 0.4, 0.5]$. Note that only every 4th hyperparameter combination is displayed on the axis for readability.

datasets, it has a huge training time (306.1 s). Due to Sinkhorn being an iterative method, the training time is substantial at 220.0 s. However, the combination of Sinkhorn with linear MMD leads to faster convergence of the loss, requiring fewer iterations and resulting in a decrease in training time (218.0 s). In terms of single loss functions, Jeffreys CORAL and Stein CORAL have a slow training time and a significant impact on the overall training time (322.2 s and 328.7 s, respectively, compared to 220.4 s for Deep CORAL).

Hyperparameter Searches on the OnHW Datasets. We perform two hyperparameter searches on the three OnHW datasets and the three encoder networks. As the TCN encoder cannot extract meaningful features, we only show results for the CNN and ResNet18 encoders. The first hyperparameter search (see Figure 2.14) trained all combinations of the parameters $drop\ last \in [False, True]$, $normalize \in [False, True]$, $kernel\ size \in [3, 5, 7]$, $stride \in [1, 2, 3]$, and $dropout\ rate \in [0.2, 0.3, 0.4, 0.5]$, as introduced in Section 2.3.2, resulting in 151,200 trainings. The optimal hyperparameters were selected as

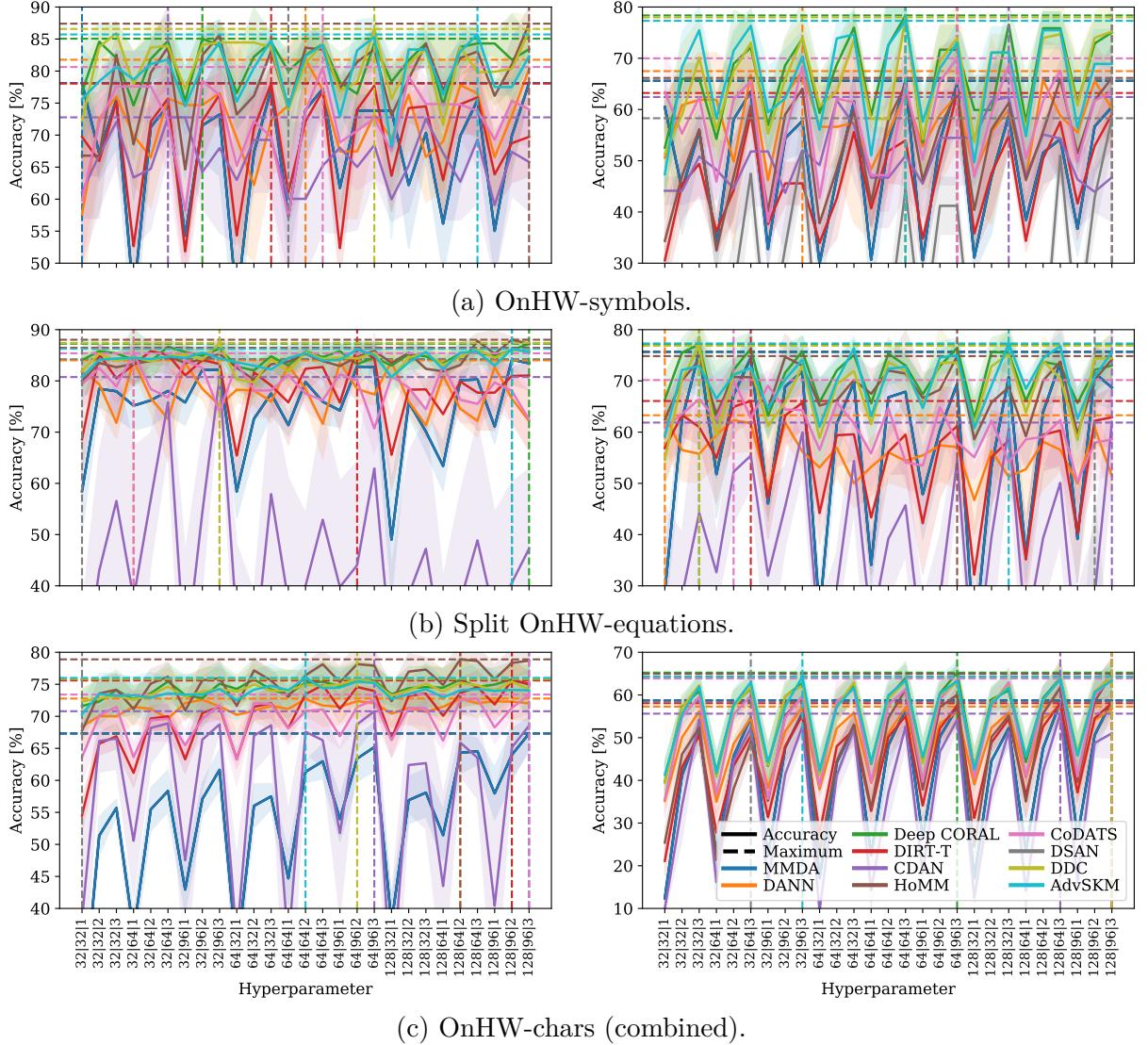


Figure 2.15: Hyperparameter search for the OnHW datasets (L→R) for the CNN encoder (left) and the ResNet18 encoder (right). Results are averaged over five runs. Hyperparameters are sorted the following: *intermediate channels* ∈ [32, 64, 128], *final output channels* ∈ [32, 64, 96], and *feature length* ∈ [1, 2, 3]. Note that we limit the y-scaling for readability.

drop last = False, *normalize* = False, *kernel size* = 5, *stride* = 1, and *dropout rate* = 0.4 for the follow-up search. For the second search, we select the parameters *intermediate channels* ∈ [32, 64, 128], *final output channels* ∈ [32, 64, 96], and *feature length* ∈ [1, 2, 3], resulting in 18,900 trainings (see Figure 2.15). The feature length significantly increases the capacity of the neural networks. We select the optimal hyperparameters as *intermediate channels* = 64, *final output channels* = 64, and *feature length* = 3, which were used for the trainings with the results shown in the Table 2.13, Table 2.14, and Table 2.15.

Part II

Online Handwriting Recognition

Chapter 3

The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning

Contributing Article

Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 4(3), article 92, pages 1–20, Cancún, Mexico, September 2020. doi:10.1145/3411842.

Publisher Website

<https://dl.acm.org/doi/abs/10.1145/3411842>

Copyright License

This work is licensed under a Creative Commons Attribution International 4.0 License (<https://creativecommons.org/licenses/by/4.0/>). © Copyright held by the owner/author(s).

Author Contributions¹⁵

This paper was a joint work between Felix Ott and Mohamad Wehbi with equal contribution, and both are the corresponding and leading authors (alphabetically ordered). Felix

¹⁵For a definition of the author contributions, see the Elsevier CRediT author statement:

<https://www.elsevier.com/authors/policies-and-guidelines/credit-author-statement>

3. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning

Ott was responsible for the conceptualization of ideas and goals, and development of the methodological design. Felix Ott supported for the software development, and did the validation, formal analysis, investigation (i.e., data collection), data curation (i.e., maintain research data), writing of the original draft (i.e., Abstract, Section 1 to 3, i.e., broad literature review and dataset description, Section 6, Figure 1, Table 1, Figure 4, Table 3, Figure 6, and Figure 7) and the review (i.e., Section 4), visualization, and project administration. Mohamad Wehbi contributed for the conceptualization of ideas and goals, software and programming of the methodology, and the creation of models. Mohamad Wehbi was responsible for the validation, formal analysis, investigation (i.e., data collection), data curation (i.e., produce metadata), writing of the original draft (Abstract, Section 1, Section 4 and 5, i.e., experiments and results, Figure 3, Table 2, Figure 5, Table 4, Figure 8, and Section 6) and the review, visualization, and project administration. Tim Hamann and Jens Barth contributed with the conceptualization, validation, investigation (i.e., data collection), resources (i.e., providing of sensor-enhanced pens for data collection), project administration, reviewing and editing, and provided images for Figure 2. Björn Eskofier contributed by supervision and funding acquisition by the Bayerisches Staatsministerium für Wirtschaft, Landesentwicklung und Energie which is part of the EINNS project (Entwicklung Intelligenter Neuronaler Netze zur Schrifterkennung) (grant number IUK-1902-0004). Christopher Mutschler contributed by supervision and funding acquisition of the project which was supported by the Federal Ministry of Education and Research (BMBF) of Germany by the research program Human-Computer-Interaction through the project “Schreibtrainer” (grant number 16SV8228). Christopher Mutschler was responsible for reviewing and editing, visualization, and the provision of resources such as computing power.

Paper Presentation

This paper was presented by Felix Ott as an IMWUT paper at the ACM International Conference on Pervasive and Ubiquitous Computing (UbiComp) jointly held with the International Symposium on Wearable Computers (ISWC) in September 2020¹⁶.

Datasets

Along this paper, the OnHW-chars dataset was published and is available at:

<https://stabilodigital.com/onhw-dataset/>

<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>

¹⁶Video Link: <https://www.youtube.com/watch?v=z043ksWRy-Y&t=1s>

Statement about Recent, Related Research¹⁷

In recent times, a multitude of studies employed handwriting devices that differ from the sensor-enhanced pen utilized in the contributing papers (Klaß et al., 2022; Ott et al., 2020b, 2022a,b,c,d, 2023c; Drey et al., 2022). These discrepancies pertain to the type of integrated sensors, the level of advancement (i.e., prototypical versus finished product), or in the intended application. Singh & Chaturvedi (2023) utilized surface electromyography signals (sEMG), accelerometers, and gyroscopes, which are – in contrast to image processing techniques – insensitive to variations in lighting conditions. In a manner akin to the contributing paper (Ott et al., 2020b), Singh & Chaturvedi (2023) employed elementary classifiers for the classification of handwritten characters that are subject to denoising autoencoder processing. However, their experimentation only entailed the recording of 26 lowercase English characters, written by 15 individuals. In their study, He et al. (2022a) employed a preliminary device that is equipped with a sensing module attached to the rear of the pen, and collected acceleration and audio data. Their proposed technique filters the acceleration data by categorizing the audio of writing, which is recorded via a microphone. The classification system is capable of distinguishing 15 classes, comprising numbers and symbols. The analogous preliminary device described in Alemayoh et al. (2022) encompasses an integrated IMU comprising an accelerometer and gyroscope, in addition to a force sensor. Their dataset incorporates 36 alphanumerical character classes, written solely by six participants. As per the findings of Alemayoh et al. (2022), the fusion of data from the IMU and force sensor results in improved classification accuracy. Evaluations indicate that a Vision Transformer model surpasses both CNN and LSTM models in terms of performance. In addition, He et al. (2022b) introduced a smart pen with triboelectric sensors that are evenly dispersed along the pen, thus enabling the tracing of handwriting trajectories as well as Latin characters and Arabic numerals. These publications collectively illustrate the significant interest within the research community pertaining to sensor-enhanced pens and OnHW recognition. Consequently, the contributions and impact of the contributing paper (Ott et al., 2020b) and its subsequent papers (Klaß et al., 2022; Ott et al., 2022a,b,c,d, 2023c) are further strengthend.

In their work, Kreß et al. (2022) introduced an approach that involves the distribution of computational workload between the sensor pen (Ott et al., 2020b) and a mobile device (such as smartphone or tablet) for handwriting recognition, due to the high system requirements that result from interference on mobile devices. Pertaining to the domain of handwriting recognition with edge devices, the study by Liu & Sugano (2022) is noteworthy. This paper outlines a technique for the efficient personalization of models on a small interactive object recognition camera device, that includes the recommendation of training candidates, a methodology that could also be extended to data collection involving sensor-enhanced pens such as Ott et al. (2020b).

¹⁷This statement encapsulates recent, pertinent research that are intrinsically associated with the subject matter of the contributing paper. Such research is disclosed subsequent to the submission or publication date of said paper and may employ the corresponding dataset(s), make reference to the contributing paper, or potentially amplify or refine the experiments outlined therein.

3. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning

Air-writing is a related but distinct area that has been widely explored with sensor-enhanced devices (Zhang et al., 2022). In future research, the sensor-enhanced pen could potentially integrate OnHW recognition with air-writing into a single device, given that both areas rely on similar sensors that are integrated into the device. Specifically, the force sensor is critical for OnHW recognition, whereas the accelerometer and gyroscope are essential for air-writing. The key variation between the two areas is the spatial scaling of hand movements.

The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning

FELIX OTT*, Fraunhofer Institute for Integrated Circuits (IIS), Nuremberg, Germany and Ludwig-Maximilians-University (LMU), Munich, Germany

MOHAMAD WEHBI*, Friedrich-Alexander University (FAU) Erlangen-Nuremberg, Germany

TIM HAMANN, STABILo International GmbH, Heroldsberg, Germany

JENS BARTH, STABILo International GmbH, Heroldsberg, Germany

BJÖRN ESKOFIER, Friedrich-Alexander University (FAU) Erlangen-Nuremberg, Germany

CHRISTOPHER MUTSCHLER, Fraunhofer Institute for Integrated Circuits (IIS), Nuremberg, Germany and Ludwig-Maximilians-University (LMU), Munich, Germany

This paper presents a handwriting recognition (HWR) system that deals with online character recognition in real-time. Our sensor-enhanced ballpoint pen delivers sensor data streams from triaxial acceleration, gyroscope, magnetometer and force signals at 100 Hz. As most existing datasets do not meet the requirements of online handwriting recognition and as they have been collected using specific equipment under constrained conditions, we propose a novel online handwriting dataset acquired from 119 writers consisting of 31,275 uppercase and lowercase English alphabet character recordings (52 classes) as part of the *UbiComp 2020 Time Series Classification Challenge*. Our novel OnHW-chars dataset allows for the evaluations of uppercase, lowercase and combined classification tasks, on both writer-dependent (WD) and writer-independent (WI) classes and we show that properly tuned machine learning pipelines as well as deep learning classifiers (such as CNNs, LSTMs, and BiLSTMs) yield accuracies up to 90 % for the WD task and 83 % for the WI task for uppercase characters. Our baseline implementations together with the rich and publicly available OnHW dataset serve as a baseline for future research in that area.

CCS Concepts: • **Hardware** → **Emerging tools and methodologies; Sensor devices and platforms**; Noise reduction; Digital signal processing; Sensor applications and deployments; • **Human-centered computing** → *Ubiquitous and mobile devices*; • **Computing methodologies** → Neural networks.

Additional Key Words and Phrases: Online handwriting recognition, character dataset, inertial measurement unit, time-series data, sensor-based pen, writer-(in)dependent, multi-stroke gestures, embedded

ACM Reference Format:

Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. 2020. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3, Article 92 (September 2020), 20 pages. <https://doi.org/10.1145/3411842>

*Both authors contributed equally to this research.

Authors' addresses: Felix Ott, felix.ott@iis.fraunhofer.de, Fraunhofer Institute for Integrated Circuits (IIS), Nuremberg, Germany, Nordostpark 84, Nuremberg, 90411, Ludwig-Maximilians-University (LMU), Munich, Germany; Mohamad Wehbi, mohamad.wehbi@fau.de, Friedrich-Alexander University (FAU) Erlangen-Nuremberg, Germany, Erlangen, 91052; Tim Hamann, tim.hamann@stabilo.com, STABILo International GmbH, Heroldsberg, Germany, Heroldsberg, 90562; Jens Barth, jens.barth@stabilo.com, STABILo International GmbH, Heroldsberg, Germany, Heroldsberg, 90562; Björn Eskofier, bjoern.eskofier@fau.de, Friedrich-Alexander University (FAU) Erlangen-Nuremberg, Germany, Erlangen, 91052; Christopher Mutschler, christopher.mutschler@iis.fraunhofer.de, Fraunhofer Institute for Integrated Circuits (IIS), Nuremberg, Germany, Nordostpark 84, Nuremberg, 90411, Ludwig-Maximilians-University (LMU), Munich, Germany.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

© 2020 Copyright held by the owner/author(s).

2474-9567/2020/9-ART92

<https://doi.org/10.1145/3411842>

1 INTRODUCTION

Handwriting involves a representation of the language by structured symbols and applies thoughts and spoken language onto paper. It is used for communication between individuals or for the documentation of thoughts for further use. Handwriting Recognition (HWR) is the process of converting written text into a digitized form that a computer can understand. HWR has been studied for several years, however, it still presents a challenge that requires further research since the need for HWR systems is growing with the extended need for the use of digitized systems [57]. This domain can be categorized into two distinguished types: offline and online HWR.

Optical Character Recognition (OCR) falls into the domain of offline HWR and describes the analysis of the visual representation making use of offline features of the input, where the input of the system is an image containing handwriting. The written paper is scanned into a digitized image through, e.g., a digitizer, a tablet or a camera, then segmented into different segments that could include lines, words, or letters, which then undergo the recognition process [56]. Offline HWR systems have reached near-human performance results and have been successfully implemented in different areas of applications such as signature verification [21], reading bank checks and postal addresses. However, offline HWR cannot be applied for applications that require a real-time recognition (as there is no image of the document immediately after the completion). Furthermore, from simply analyzing the images it is not possible to make use of rich information such as the temporal direction of writing, the writing order, writing speed, and (in some cases) the pressure of writing. Only using the position of the strokes leads to ambiguities if letters overlap [15, 71].

Online HWR (OnHWR) typically uses time in association with different types of spatio-temporal signals. The data may contain a form of positions including information about the displacement of certain input devices, or may include the movement of the input devices on the writing surface. These signals are then processed by a recognition system that orders the strokes by their position and time and that can make use of the geometrical design and dynamic information from the movement of the writer. In many previous work a stylus pen together with a touch screen surface usually serve as input devices. Through the temporal information online HWR systems can be more accurate than offline systems, since similarly shaped characters can be distinguished by knowing the number of strokes that were necessary [56, 71].

One application of HWR systems is the commitment in primary school classes, where the teacher instructs an essay, for example, the pupils write with the sensor-enhanced pen on normal paper, and the text can be converted to a computer-based format automatically and online. The teacher receives immediately a status of the process. Furthermore, the written text can directly be corrected, and decreases the teacher's effort. Currently, no HWR system suffices all requirements for such an application, as such systems are either offline, require a pen that influences the graphomotoric of the writer, or requires for writing on a tablet that is expensive and influences the writing style [48]. The required device for the sensor-pen is just a computer, tablet, or phone with an installed app with a pen-device bluetooth connection that is often available anyways.

For the evaluation of HWR systems and also of the writing-style writer-specific and platform-specific aspects are necessary that have to be considered. The identification of handedness of the writer plays an important role to study and compare left-handed and right-handed writers. Previous work analyzes the handedness on the basis of strokes and slope of letter [63]. The writing performances of dysgraphic and proficient writers are compared by a distinction between the number and duration of two kinds of pauses, i.e., pen stops and pen lifts [55]. For the design of the recording platform pen-based systems can be favored over tablet or keyboard systems, as writing with a pen provides better cognitive processing, i.e., theoretical understanding, critical thinking and memory recollection [3, 65]. Modifications in writing conditions, e.g., a keyboard or a smoother writing surface of a tablet, might influence the writing performance, in particular, those of non-automatized beginning writers such as children as their handwriting movements require visual and graphomotor feedback [26]. Hence, pen-based OnHWR systems on paper have the lowest impact on the graphomotor.

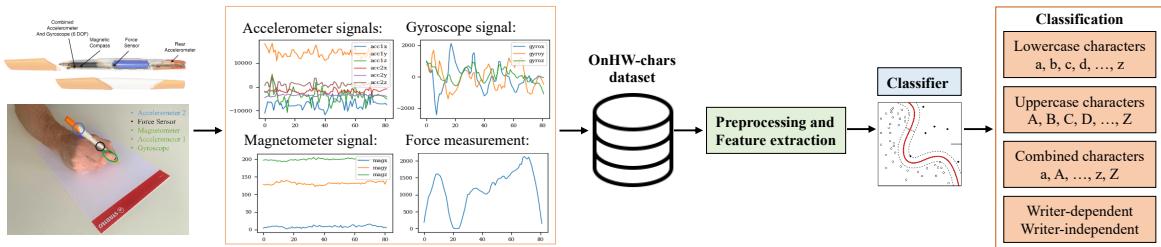


Fig. 1. Complete pipeline. (1) The recording setup is a STABILO DigiPen and a tablet storing sensor data and ground truth labels. (2) The ballpoint pen is enhanced with two accelerometers, one magnetometer, one gyroscope and one force sensor. (3) The OnHW-chars database consists of 31,275 uppercase and lowercase letters (52 classes) from 119 writers. (4) Pre-processing and noise filtering is necessary for (5) training the classifier. (6) We present the results for lowercase, uppercase and combined letters, both on writer-dependent (WD) and writer-independent (WI) classification tasks.

HWR systems require large amounts of training data to acquire the ability of understanding and classifying what the user is writing. However, the data collection process is a time and resource consuming process. Consequently, for the sake of progressing within the specific domain of research, collected datasets are shared within the scientific community. This field has been researched for many years with several databases being published. However, many of these published datasets were collected using specific expensive equipment [1, 47, 51] that make a recognition system unachievable when applying for a real use-case utilization [45], are too small [77], or only address specific aspects [11, 34, 60]. Hence, the availability of a dataset collected with a convenient digital pen is essential for the scientific community. The primary purpose underlying our research is to implement a HWR system that uses a digitizer in the form of a pen that transmits data online during the writing process. To train such a recognizer for efficient recognition, we need a sophisticated dataset.

The main contribution of this paper is to share a large dataset adding a scientific value in the handwriting recognition domain. We present a dataset of alphabet characters written on plain paper in the form of time-series data collected from a digital ballpoint pen equipped with sensors, i.e., a STABILO DigiPen. The collection of data written on normal paper makes it easier to apply a writing recognizer without the need of other more expensive devices or specific writing surfaces. In addition, we implemented (most of) the previously used methods applied for OnHWR, i.e., Machine Learning (ML) classifiers such as k-Nearest Neighbour (kNN) and Support Vector Machine (SVM) and also Deep Learning (DL) methods such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), i.e., Long-Short-Term Memorys (LSTMs), on our dataset. This provides a solid baseline for future research and fosters reproducibility in this research area. Fig. 1 presents the whole pipeline including data recording, pre-processing, classifier training, and evaluation.

The remainder of this paper is structured as follows. Section 2 discusses similar datasets that are currently used in the field. A summary of available offline and online datasets in the handwriting recognition domain is provided, including the type, recording platform and size of data. Section 3 presents our main contribution: a novel dataset for online handwriting recognition. We present the digitizer used for data acquisition along with the detailed description of the collected dataset. Section 4 provides methods used to pre-process data, to extract features from it, and how to apply online recognizers, i.e., we describe our implementation of the ML and DL baseline models that aim at solving the classification problem that this dataset offers. We provide quantitative results and discuss them in Section 5. Section 6 concludes.

2 RELATED WORK

Over the last decade, online handwriting recognition has shown promising results and high accuracy. The number of datasets and methods to evaluate HWR systems steadily increases. While offline HWR is already very advanced [14, 49, 74], the focus of research moves to gesture recognition [4, 25, 38, 43, 69, 76], human activity recognition [79], and online HWR systems [19, 35, 36, 39, 73].

To train an OnHWR system a dataset needs to fulfill a number of requirements. (1) The dataset should allow for both a WD and WI evaluation, as writing recognition has to be applicable to new writers without re-training the model. Here, the difficulty is to cover many writing styles, i.e., printed or cursive writing, holding of the pen, pressure of the pen, size of the characters, and influence of noise. For that, the age, gender, education and frequency of writing has to be distributed homogeneously over the dataset. (2) As the number of alphabet classes is high and the introduced noise of the sensors can vary to a high degree, we need a large number of writers to guarantee for an optimal learning procedure to be considered for the later evaluation of the classifiers. (3) Finally, the dataset must contain time-series data for online recognition captured at a high frame-rate in order to allow for high classification accuracies.

This section provides a review of offline and online recognition datasets in Sections 2.1 and 2.2 with a focus on the requirements on the dataset for an optimal online writing recognition. To better compare the available dataset we line out a summary in Table 1 which summarizes all offline and online datasets, their corresponding recording platform, the size of the data, and the corresponding evaluation methods. Due to the broad spectrum of associated applications, the diversity of patterns, and our main contribution on OnHWR systems, we split these datasets into digits, characters and words, gestures, shapes and symbols datasets.

2.1 Offline Datasets and Recognition Systems

The development of offline datasets started in early 1900's. The IAM [49] dataset is one of the most commonly used dataset and provides English words and sentences. The large NIST dataset [22, 75] and its variants SD-19 [29], MNIST and EMNIST [14] contain digits and characters, but suffer from high ceiling effects, i.e., less generalization leads to overfitting. Further datasets cover addresses, e.g., the CEDAR [33] dataset, and outdoor image texts, e.g., the SVT [74] dataset. More offline datasets are listed in Table 1. The classes of our OnHW-chars dataset are the same as the classes from the IAM [49] dataset, but the dataset was acquired on a whiteboard and not on paper. The OnHW-chars dataset is smaller in size compared to the NIST and MNIST datasets, but larger as other visual image datasets [49, 74].

Existing recognition systems differ regarding pattern representation (i.e., image templates, structural representations and feature vectors), drawing constraints, and decision-making processes. The datasets present a large diversity of content with very different properties. We differentiate the datasets between their number of classes, the available amount of training samples per class, and between WD and WI experimental settings. The recognition of some datasets are quite challenging because of the presence of different writing styles and noisy data, while some datasets enable an easier recognition [17]. Most of previous recognition systems focus on writing on an electronic device. This requires an expensive device, and the recognition system cannot be used on normal paper. Hence, we focus on pen-based recognition systems that have integrated sensors.

2.2 Online Datasets

In the following, we describe datasets that are a collection of digits, characters and words more related to our OnHW-chars dataset. The LaViola [40] dataset has been written by 11 persons with a pen on a *TC 1100* tablet and covers trajectory-based digits, characters and mathematical symbols. They used an AdaBoost classifier and yield an accuracy between 90.9 % and 97.19 % for different recognizer configurations. Keshari et al. [37] achieved

Table 1. Overview of state-of-the-art offline and online pen-based handwritten datasets. The writer-dependent (WD) and writer-independent (WI) column indicate the possibility of running WD and WI experiments. As we focus on OnHWR systems, we do not declare reported experiments for offline datasets.

Dataset	Classes	Recording platform	Size writer	Size sample	Size class	WD/WI	Experi- ments
Offline datasets [78]							
NIST [22, 75]	Handwritten digits	Pen	3600	800,000	10	n.d.	n.d.
MNIST	Subset of NIST	Pen	–	70,000	10	n.d.	n.d.
EMNIST [14]	Digits and letters	Pen	–	445,600	36	n.d.	n.d.
Mathematics	Mathematical symbols	Pen	–	60,000	10	n.d.	n.d.
Devangari	Devangari characters	Pen	25	1,800	36	n.d.	n.d.
Arabic Text	Lexicon of words	Pen	–	113,284	–	n.d.	n.d.
Document	Lists, tables, formulas, diagrams and drawings	Handwritten documents	189	941	–	n.d.	n.d.
CEDAR [33]	Characters and digits	Pen	1,500	59,584	–	n.d.	n.d.
CENPARMI	Digits	Pen	–	–	–	n.d.	n.d.
IAM [49]	English word, sentences	Pen on a whiteboard	657	115,320	1,539	n.d.	n.d.
Street View Text (SVT) [74]	Outdoor image text from businesses	Harvested from Google street view	–	725	–	n.d.	n.d.
Online and gesture-based datasets (see Section 2.2) [16]							
Digits, characters and words datasets							
OnHW-chars	English characters	Sensor-enhanced pen	119	31,275	52	WD/WI	–
UNIPEN [30]	Latin alphabet, characters, words and sentences	Pen-based computers	–	12,000	–	–	–
PenDigits [1]	Handwritten digits	Wacom PL-100V	44	10,992	10	WD/WI	[1]
UJIpenchars / UJIpenchars2 [47]	Isolated handwritten characters	Toshiba Portégé M400 tablet PC	11	1,364	62	WD/WI	–
LaViola [40]	Digits, characters and math symbols	Tablet TC 1100 with pen	60	11,640	97	WD/WI	–
IME-OnDB [11]	Letters and gesture	Tablet TC 1100 with pen	11	11,602	48	WD/WI	[17, 37, 40]
IAM-OnDB [45]	Word instances	Pocket PC with pen	14	6,636	18	WI	[12, 18]
Gesture datasets							
Match-Up & Conquer [58]	Multi-touch gestures	Multi-touch display 3MTM C3266PW	16	5,155	22	WD/WI	–
NicIcon [51]	Gestural commands	Wacom Intuos2 A4	34	26,163	14	WD/WI	[7, 17, 68, 76]
Sign-OnDB [2]	Single-stroke pen gestures	Tablet with pen	20	33,150	17	WD/WI	[17, 25, 43]
unistroke [77]	2D single-stroke gestures	HP iPAQ h4355 with pen	10	4,800	16	–	[5, 44, 50, 67]
MMG [6]	2D multi-stroke gestures	Finger or pen on tablet	20	9,600	16	WD/WI	[6, 69]
Multitouch gesture [59]	Multi-touch symbolic gestures	Multi-touch display 3MTM C3266PW	18	7,200	30	–	–
ILGDB [60]	Single-stroke pen gestures	Tablet with pen	38	4,656	588	WD	[17, 43]
UsiGesture [9]	Gestures	Tablet with pen	30	18,300	61	WD/WI	[8–10]
Objects, shapes and symbols datasets							
HBF49 [17]	Features	Written with online device	–	–	49	WD/WI	[17]
Object sketches [20]	Object sketches	Multi-strokes	1,350	20,000	250	WD/WI	[41, 42]
HHReco [32]	Geometric shapes	Wacom Graphire2 pen	19	7,791	13	WD/WI	[17, 52, 53]
CVCsymb [62]	Architectural and electrical symbols	Digital pen	25	5000	50	WD/WI	–
IMISketchSDB [34]	Offline architectural symbols	Architectural plans	50	1,871	13	WI	–
HOMUS [13]	Online music notations	Galaxy Note with SPen	100	15,200	38	WD/WI	–

an accuracy up to 94.57 % on the mathematical expressions utilizing SVMs trained on standard and Chebyshev coefficient features.

The UJIpenchars/UJIpenchars2 [47] datasets contain 62/97 different classes of characters and symbols recorded by 11/60 writers on the *Toshiba Portégé M400 tablet* covering 1,364/11,640 samples. UNIPEN [30] is an ongoing project of collecting handprint and cursive handwriting on a pen-based computer from various alphabets including Chinese and Latin, pen gestures and signatures. The sentences dataset IAM-OnDB [45] covers about 86,272 word instances from an 11,059 dictionary written by 221 writers via an electronic interface from a whiteboard. Their Hidden Markov Model (HMM) based approach achieves 65.9 % accuracy. The IME-OnDB [11] dataset is a good benchmark for evaluating relative positioning of handwriting, as several subsets of gestures have the same shape only distinguishable through their spatial context. To take relative positioning into account [11] exploits a fuzzy approach. The PenDigits [1] dataset is a collection of digits written by 44 users on the *Wacom PL-100V*. 10,992 samples covering the 10 digit classes and allow WD and WI experiments. Through the combination of static and dynamic Multi-Layer Perceptron (MLP) classifiers the accuracy can be increased by different fusion techniques, i.e., voting, mixture, stacking, boosting and cascading. The accuracy of the combined classifier dropped from 99.3 % for the WD testing set to 98.3 % for the WI testing set (see similarity to our results in Section 5).

The following datasets build a database for human gestures and are more related to human computer interaction. The NicIcon [51] dataset contains 26,163 of offline and online written iconic multi-strokes gestures (emergency situations, e.g., accident, fire), and includes pen-up movements and pressure measures. Through a highly varying order and number of strokes, the dataset is quite noisy. WD and WI experiments exist: fusion of HMM-based and Zernike methods [7], a CKMeans with auto-completion algorithm [68], and MLP, SVM and Dynamic Time Warping (DTW) classifiers using global and stroke-level features [76].

For the Sign-OnDB [2] dataset 33,759 samples of single-stroke gestures are collected from 20 persons written on a tablet. Some of the 17 classes can only be distinguished based on dynamic information [17]. The ILGDB [60] dataset is a collection of single-stroke gestures recorded with a tablet. Each of the 28 writers provided 21 different gestures of their choice, and hence, in this dataset exists a large number of different samples unequally distributed over the classes. Consequently, only WD experiments exist [43]. The Match-Up&Conquer [58] multi-touch dataset is designed to address how users articulate gestures. Similar is the Multitouch gesture [59] dataset that covers 7,200 samples from 18 participants. 30 different gesture classes, e.g., circle, triangle, heart and cat, are unique in the number of strokes of the shape, number of fingers touching the surface, and bimanual or single-handed inputs. The unistroke [77] dataset consists of 16 different 2D single-stroke gestures, e.g., triangle, question mark and start. This dataset is evaluated by the \$1 [77] and \$N [5] recognizer, protractor [44], and DTW [50, 67]. The \$N-Protractor [6] is derived from the \$1 unistroke [77] recognizer that uses a closed-form template-matching method instead of an iterative search method in the \$N [5] recognizer. They provided the Mixed Multistroke Gesture (MMG) [6] dataset representing 16 classes of 2D multistroke gesture symbols. UsiGesture [9] is a software support platform that accommodates multiple algorithms for pen-based gesture recognition. The goal is a dataset made of characters, symbols and commands, that allows to evaluate a gesture recognition algorithm depending on contextual variables, e.g., environment, platform and user.

Objects, shapes and symbols are addressed in the following datasets. The CVCSymb [62] dataset is a combination of online and offline architectural and electrical symbols. 5,000 samples have been drawn by 50 writers separated in two groups of 25 writers each. This results in total in 50 WD and WI classes. The HHReco [32] dataset consists of 7,410 samples in total of 13 different geometric shapes, i.e., circles, cylinder, archs, and polygons, written by 19 people on a *Wacom Graphire2* tablet. Ouyang et al. [53] use an image deformation model to achieve 98.2 % accuracy focusing on the visual appearance of the symbols.

The Object sketches [20] dataset is a collection of 20,000 unique sketches, e.g., teapot and car, evenly distributed over 250 object classes. They built upon a bag-of-features representation to extract local features

and construct a visual vocabulary using kMeans clustering to train a SVM classifier, and achieved 56 % accuracy. Related is SHREC'13 [41] and SHREC'14 [42] that refer to sketch-based 3D shape retrieval containing 7,200/12,680 sketches and 1,258/ 8,987 3D models. The IMISketchSDB [34] dataset is an offline collection of 13 different architectural symbols, e.g., furniture, covering 1,871 samples from 50 plans. HOMUS [13] is the only dataset that addresses musical symbols, which consists of 15,200 offline and online samples from 100 users covering 32 symbol classes. kNN, DTW and HMM are used for the online classification task, and kNN, NN, SVM and HMM are used for the offline classification task.

HBF49 [17] is a unique set of 49 features focusing on the universal representation space adapted to a large variety of symbols that can be used as a reference for evaluation of symbol recognition systems. They reported experiments with 1NN and SVM classifiers for eight different datasets [2, 32, 34, 40, 51, 60, 62, 70] for WD and WI cases.

A similar recording platform to the STABILO DigiPen is a phone with gyroscopes and accelerometers used in [19]. The GyroPen method reconstructs the trajectory of the phone's corner touching a writing surface for pen-like interactions. An online recognition system is used using an extended feature set to recognize the words with trajectory coordinates as input. Neelasagar et al. [35] also use the accelerometer and gyroscope signals from a smartphone for 3D handwritten character and gesture recognition. The acceleration signals are pre-processed with segmentation, filtering and normalization, while the gyroscope signals are lowpass filtered and normalized to get the orientation correction of the device.

Inertial pens that are closest to ours are the ones used in [36, 39], but the recorded dataset is not published. In these publications, accelerometer, gyroscope and magnetometer are integrated in a pen along with a microcontroller and a wireless transmission module that records movement data for writing alphabets and making gestures [36]. Unfortunately, the data acquisition unit is a large device that influences the style of the writer. Statistical features gave the best results in combination with a probabilistic NN and a kNN classifier. A similar device was constructed by [73] The recorded acceleration signals are calibrated, lowpass filtered, segmented and normalized, before aligning the signals with the 10 digit classes by a DTW method. WD (90.6 % accuracy) and WI (84.8 % accuracy) experiments are reported that are in the same range as our experiments on our OnHW-chars dataset (see Section 5).

Koellner et al. [39] use the STABILO DigiPen, which is the same device we used ourselves for the recording of the OnHW dataset, but their dataset is not published. The dataset consists of 20,000 English lowercase letters written from 15 users, and hence, created a dataset with more samples per writer per class than our dataset. WD and WI results for kNN, LDA, NB and LSTM classifier are reported.

The recording platform of most of the online datasets are pen-based computers [11, 30, 62], tablets [1, 2, 6, 9, 32, 40, 47, 51, 58–60], phones [13, 77], or a whiteboard [45]. The classes of only some of these datasets [1, 11, 30, 40, 47] are similar to our OnHW-chars dataset, i.e., gesture-based [2, 6, 9, 51, 58–60, 77] and object/symbol-based [13, 17, 20, 32, 34, 62] classification are related to other applications. As many other datasets, WD and WI is possible on OnHW-chars, but a large number of writers is necessary to evaluate for that in detail. While IAM-OnDB [45] (221) and object sketches [20] (1,350) have higher, all other datasets have lower number of writers than OnHW-chars. Similar to [2, 45, 51] OnHW-chars is in the upper scope of number of samples (31,275).

3 PROPOSED DATASET

This section introduces our novel online handwriting (OnHW) dataset. We present our IMU-sensor enhanced ballpoint pen, i.e., the STABILO DigiPen, in Section 3.1, and describe the data acquisition constraints and calibration aspects in Section 3.2. In Section 3.3, we present our novel OnHW-chars dataset in detail.

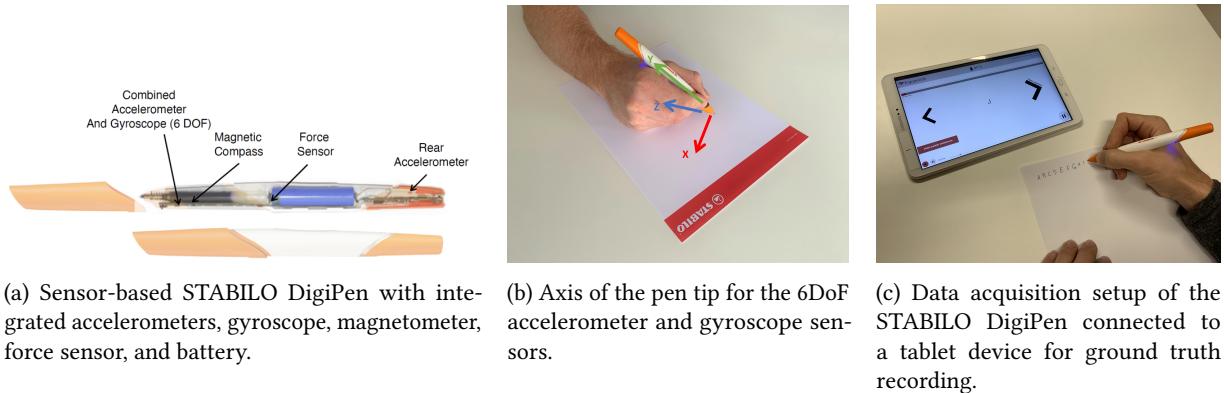


Fig. 2. The STABILO DigiPen.

3.1 Sensor-Enhanced Ballpoint Pen

The STABILO DigiPen is a sensor-enhanced ballpoint pen with internal data processing capabilities, see Fig. 2a. A Bluetooth module enables live streaming at 200 Hz to a connected device. The pen's overall length is 167 mm, its diameter is 15 mm, and it weighs 25 g. With its ergonomic soft-touch grip zone it is easy to use and feels comfortable and natural. Each DigiPen is equipped with a front accelerometer (STM LSM6DSL), a gyroscope (STM LSM6DSL), a rear accelerometer (Freescale MMA8451Q), a magnetometer (ALPS HSCDTD008A), and a force sensor (ALPS HSFPAR003A) [66].

The data recordings store 14 measurements provided by the sensors: two acceleration, one gyroscope and one magnetometer signals (each in X, Y, and Z direction, see Fig. 2b), the force with which the pen tip touches the surface, and the timestep at which the tablet receives the data from the pen.

3.2 Data Acquisition and Calibration

We use a recording app provided by STABILO International GmbH to obtain the sensor data, which is connected to the DigiPen and tells the user which character to write (see Fig. 2c). Through this setup we also record the ground truth labels. We applied the following constraints for our data recording to achieve a homogeneous and equally distributed dataset. The writer has to sit on a chair in front of a table, and has to write on a normal, white paper (80 mg/m²) padded by five additional sheets. There was no guideline concerning the size of the handwriting and the way of holding the pen, just the logo needs to face upwards. Users are allowed to write in a printed and cursive style.

Prior to recording we need to calibrate the pen with a short two-step procedure to determine the gyroscope and magnetometer biases and the magnetometer scaling. While placing the pen on the table for a couple of seconds, it is possible to find the gyroscope biases bg_x , bg_y and bg_z for each axis as the gyroscope values are supposed to be zero. Then, the pen should be rotated in all directions (covering a sphere). With the cloud of magnetometer points, we can calculate the sensor's bias bm_x , bm_y and bm_z (the sphere's center) and the sensor's scaling factor sm_x , sm_y and sm_z (the sphere's radii). More information can be found in [54, 61]. With these values, the raw values can be scaled and the bias removed. For each sensor the SI value without bias SI_{bias} can be computed with

$$SI_{bias} = \frac{raw_{value} - bias}{\frac{\max}{\min} sm_*}, \quad (1)$$

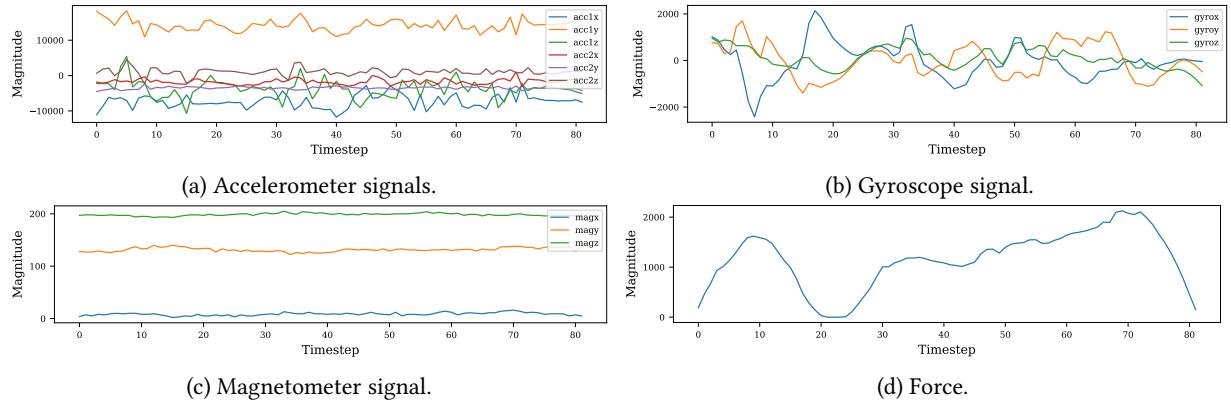


Fig. 3. Exemplary sample of the uppercase character 'B' for the front and back accelerometer (a), gyroscope (b), magnetometer (c), and force sensor (d).

where raw_{value} is the measured value from the datasheet sensor, $bias$ is the bias from the calibration procedure (bg_x , bg_y and bg_z for the gyroscope, bm_x , bm_y and bm_z for the magnetometer, and 0 for the accelerometers and the force sensor), max is the maximal range value that is 32,768 of the front accelerometer and the gyroscope, 8,192 for the back accelerometer and the magnetometer, and 4,096 for the force sensor, max_{SI} is the maximal SI value that is 2 g for both accelerometers, $1,000 \text{ }^\circ \text{s}^{-1}$ for the gyroscope, 2.4 mT for the magnetometer, 5.32 N for the force sensor, and sm_* is the measured scaling factor for the corresponding axis, otherwise it is 1 [66]. With the calibration procedure from [61] the bias of the accelerometer cannot be determined, and hence, we set it to 0. We use the raw data in Section 4.

Fig. 3 shows exemplary raw signals of a written character 'B' (note that there is a total number of 82 timesteps). As the letter is constructed of two strokes, the pen is lifted one time and the measured force is 0 N between timestep 21 and 24, see Fig. 3d. We describe a proper pre-processing of such signals in Section 4.1.1.

3.3 The OnHW-chars Dataset

In the future, our OnHW dataset consists of several sub-datasets. In this paper, we first provide the OnHW-chars dataset that consists of lowercase and uppercase characters. The recording of further datasets is an ongoing project and will be continuously increased for a more profound and detailed evaluation. Words, sentences, symbols and numbers from the same writers will be published in a later stage. Our DigiPen records data measurements at 100 Hz. For each writer, three .csv files are provided: One file that contains the calibration data, one file that contains the character labels with the start and end timesteps, and one file that contains the 13 measurements for each timestep. The OnHW dataset is publicly available for download here: <https://stabilodigital.com/onhw-dataset/>.¹

For the novel OnHW-chars dataset 119 right-handed persons wrote the English alphabet for six times both in lowercase and uppercase letters. All writer are grown-up and above the age of 18, but the exact age was not reported due to anonymity. The ratio between women and men is 45 % women and 55 % men. This allows to solve classification problems with 52 classes. In total, this resulted in 312 samples per person (with some small deviations). An overview of the sample numbers are given in Table 2. There are 15,650 lowercase characters and 15,625 uppercase characters. The complete OnHW-chars dataset consists of 31,275 samples in total. Consequently, the OnHW-chars dataset is a large dataset of 119 writers to evaluate for a large diversity of properties, i.e., different

¹ Alternative download link: <https://iis.fraunhofer.de/onhw-dataset/>

3. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning

92:10 • Ott and Wehbi, et al.

Table 2. Overview of the number of samples of the OnHW-chars dataset including lowercase and uppercase characters from 119 writers for a writer-dependent (WD) and writer-independent (WI) evaluation.

Dataset	Total		Samples WD/WI	
	Writer	Samples	Training	Testing
Lowercase characters	119	15,650	11,542	4,108
Uppercase characters	119	15,625	11,517	4,108
Total	–	31,275	23,059	8,216

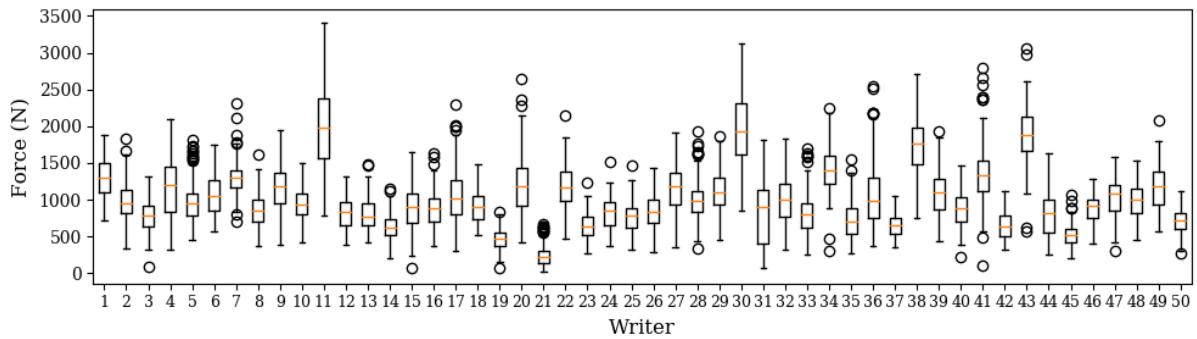


Fig. 4. Analysis of the writing properties of 50 participants. We provide the distribution of the force averaged over each sample.

writing style (e.g., printed or cursive characters), holding of the pen, pressure of the writer on the pen, and influence of noise (e.g., bias and scaling) on the classification accuracy.

As OnHW-char is an online dataset it is possible to evaluate for time-series based recognition methods, i.e., incorporate the drift of the sensors. Constructing a WI recognizer is a much more challenging task as constructing a WD recognizer. However, many applications only allow for WI recognizers, as the application does not allow for a re-training to a new writer before using the pen in many cases. For the WD evaluation we split the dataset in 90 recordings for training (23,059 samples in total), and 29 recordings for testing (8,216 samples in total). This corresponds to a split of 73.73 % for training and 26.27 % for testing for the WI case (see Section 5). To better evaluate for the WD and WI tasks, Fig. 4 shows writing properties for 50 different writers, e.g., the writers 11, 30 and 43 put high pressure on the pen, while the writer 19, 21 and 45 put very low pressure on the pen.

Table 3 presents an analysis of the character properties. If the number of timesteps and strokes are highly different between the characters, the features of such samples might be better separable for ML-based classifiers. Obviously, the trajectory for uppercase characters is longer, and consequently, the average timesteps (TS) the writer requires for lowercase characters are 44.1, while the average timesteps for uppercase characters are 52.1. For example, the characters 'B' (71.2), 'E' (74.2), 'F' (67.8) and 'H' (65.6) require more time to write than, e.g., 'c' (26.8), 'l' (27.1) and 't' (27.7), as they are constituted by more strokes. Lowercase characters are constituted of 1.24 strokes on average, while uppercase characters are constituted of 1.50 strokes, e.g., the characters 'c', 'h', 'o', 'r', 's', 'v' and 'w' are always written in one stroke. The standard deviations D_{TS} and D_S indicate a high difference in writing style of a character, e.g., the stroke deviation is 0.44 for lowercase and 0.69 for uppercase characters on average.

Table 3. Analysis of the character properties. Presented are the average number of timesteps (TS) and strokes (S) and their deviations (D_{TS} , D_S) for every character.

Char.	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
TS	45.4	45.4	26.8	50.7	39.5	49.2	53.9	41.6	40.1	53.7	53.9	27.1	58.3	40.0	33.7	50.8	53.7	32.4
D_{TS}	32.6	28.7	21.5	29.7	27.8	18.1	23.3	17.7	21.6	24.7	27.6	21.3	33.5	20.8	17.5	37.3	25.5	16.6
S	1.07	1.04	1.01	1.12	1.03	1.58	1.03	1.01	1.86	1.91	1.52	1.01	1.06	1.01	1.01	1.15	1.21	1.01
D_S	0.63	0.24	0.12	0.43	0.43	0.66	0.25	0.11	0.71	1.07	0.74	0.18	0.51	0.13	0.19	0.48	0.51	0.12
Char.	s	t	u	v	w	x	y	z	A	B	C	D	E	F	G	H	I	J
TS	36.0	46.7	37.7	33.2	50.8	44.2	48.2	52.8	60.9	71.2	30.4	56.2	74.2	67.8	56.3	65.6	27.7	45.6
D_{TS}	20.2	19.2	19.9	19.2	28.1	21.5	39.9	26.7	30.9	25.2	21.9	26.9	26.1	36.6	25.8	22.7	27.5	25.9
S	1.01	1.81	1.04	1.01	1.00	1.79	1.35	1.62	1.70	1.61	1.09	1.73	2.73	2.53	1.18	2.45	1.10	1.06
D_S	0.18	0.68	0.47	0.13	0.07	0.73	0.94	0.65	1.13	0.66	1.55	0.77	1.27	1.15	0.48	1.19	0.50	0.46
Char.	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
TS	59.7	33.4	60.7	55.1	37.1	52.4	65.3	63.6	39.5	47.5	40.1	35.4	58.1	47.8	52.0	60.0		
D_{TS}	26.5	20.1	22.5	29.9	22.7	22.4	24.3	28.4	31.3	19.7	24.2	18.6	35.1	23.0	38.2	31.4		
S	1.83	1.04	1.21	1.27	1.01	1.43	1.86	1.39	1.01	1.85	1.01	1.01	1.08	1.81	1.58	1.64		
D_S	0.93	0.38	0.56	0.61	0.11	0.62	0.71	0.61	0.13	0.70	0.11	0.12	1.21	0.64	0.72	0.69		

Classifying characters from right-handed and left-handed writers from one single signal-based dataset is a quite challenging task as the pen rotation is significantly different. Hence, we decided to exclude left-handed recordings for now.

4 PROPOSED BASELINE CLASSIFIERS

There is an exhaustive literature that deals with the classification of characters, gestures, symbols and objects gathered from a 2D tablet-based recording platform. Popular methods include Dynamic Time Warping (DTW) [13, 50, 51, 67, 73, 76, 77], k-Nearest Neighbors (kNN) [13, 17, 20, 36, 39, 60], Support Vector Machines (SVMs) [17, 20, 32, 37, 51, 60, 76], Hidden Markov Models (HMMs) [7, 13, 28, 45, 46] and Neural Networks (NN) [1, 1, 28, 36, 39, 46, 76]. Indeed, research that addresses a signal-based handwritten text analysis gathered from a digital pen comparable to our platform is very rare.

Online character recognition is based on the analysis of a given sequence of strokes applied over time. Such analysis usually pre-processes the input signals (by noise filtering), and then extracts features that allow for a recognition of written characters. In this section we present these steps and apply character classification over the proposed dataset. Given the unavailability of any previous results of classifying the complete alphabet letters, we run the following experiments over the separated uppercase and lowercase letters, hence classifying 26 different classes. In addition, we present the results of applying the classifiers over the complete 52 character classes. We present results for classical ML models in Section 4.1, and for DL models that use the raw input data to classify the written characters in Section 4.2.

4.1 Character Classification using Classical Machine Learning Models

We implemented pre-processing steps for applying different ML algorithms to evaluate how accurately different models classify the alphabet characters. As a pre-processing step we applied noise filtering to reduce the noise within the data (see Section 4.1.1). Using the filtered data, we extracted different features (see Section 4.1.2), and used an autoencoder for automatic feature extraction (see Section 4.1.3) as a representation of the information in the data to be used in the classification algorithms (see Section 4.1.4).

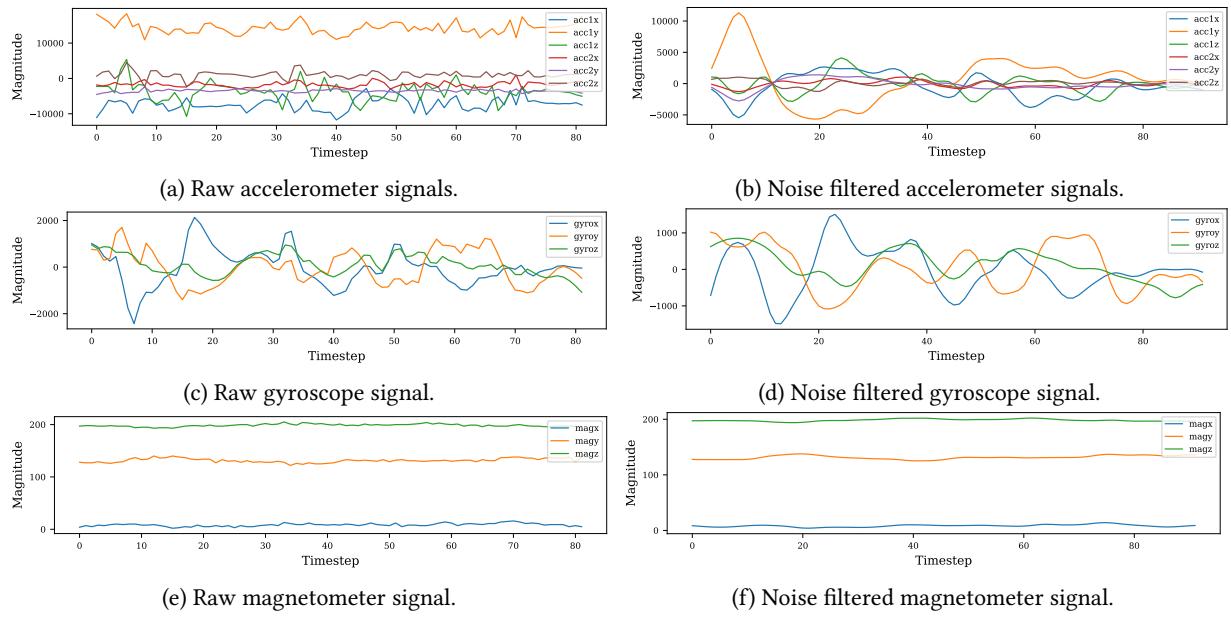


Fig. 5. Comparison of raw sensor signals (a,c,e) and noise filtered signals (b,d,f).

4.1.1 Pre-processing. Sensor noise represents the random variation in its output when functioning under static conditions. Hence, our raw sensor data output usually contains distorted signals (due to the paper surface inconsistency and trembling during writing). Pre-processing helps to remove insignificant or redundant information, which helps to extract high quality features from the signal streams. Commonly used pre-processing techniques include resampling, normalization, segmentation, and filtering [1, 35, 39, 45, 48, 72, 73]. More sophisticated approaches such as Butterworth filters and Savitzky-Golay pre-processing have been used in [23, 24, 39].

We apply a high pass filter with a cutoff frequency of 1 Hz to remove the gravitational acceleration from the accelerometer recordings. Gravity is a constant force and the high pass filter allows us to keep the fast changing forces applied when recording while filtering the slow changing gravitational force. To disregard the noise within the raw data, we use a moving filter with a window of size 11, which acts as a low pass filter that allows the removal of high frequency noise from the input data. Since random noise usually includes random jumps in the data signals, the filter allows signal smoothing. Fig. 5 shows a recorded letter sample signal before (left) and after (right) pre-processing.

4.1.2 Manual Feature Extraction. Feature extraction is the concept of deriving a new set of inputs from the original raw dataset that represents valuable information of the data in a format that best fits an ML algorithm. Well established statistical features include the mean, standard deviation, variance, mean absolute deviation, location of zero crossings, signal range, and minimal and maximal values. Fast Fourier Transform (FFT), Autocorrelation Function (ACF) and Wavelets (WFLT) are used in [39]. For trajectory-based classification techniques static features (box aspect ratio, length, curvature, area of convex hull, closure, perpendicularity, ratio of the principal axes, etc.) and dynamic features (initial angle, position of first and last points, etc.) are important [51, 60]. Furthermore, pressure data available in online data is also important, i.e., average pressure and pen down count [51]. For a very good overview and discussion on different features we refer the reader to [17].

In our system and dataset we use the two accelerometers and the gyroscope to extract multiple time and frequency domain features that would allow a higher recognition rate. We extract the features per channel

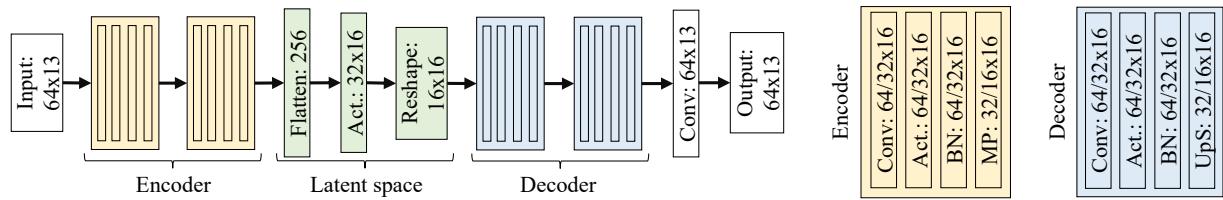


Fig. 6. Network structure of the Autoencoder. The input is encoded and the latent space representation decoded. Layers: Conv = Convolution; Act. = Activation; BN = BatchNormalization; MP = MaxPooling1D; UpS = UpSampling1D.

and concatenate the resulting feature vectors forming the final feature vector that is used for the character classification. The extracted features are mainly statistical and geometrical features of the raw signals. For the time domain features we used the maximum, the minimum, the mean value, the standard deviation, and the correlation coefficients of each of the axes. We also include the skewness (i.e., that describes the lack of symmetry of the data distribution), interquartile range (i.e., a measure of statistical dispersion), median absolute deviation (i.e., a measure of deviation from the median of the data), and area under curve. For the frequency domain features, we apply Fast Fourier Transform (FFT) to compute the Discrete Fourier Transform (DFT), then extract the previously stated features, in addition to including the weighted mean of the frequency distribution, the DFT coefficients, the local maxima of DFT coefficients, and their corresponding frequencies. The final feature vector is composed of 327 features of the concatenated channel feature vectors.

4.1.3 Automatic Feature Extraction. As there is no best practice on standard features that are usually considered best for online character classification, we also investigate the use of an autoencoder to automate the extraction of a feature vector as the extraction of manual, hand-crafted features has its drawbacks. The number of features to extract, the relation between the extracted features, and which specific features are useful for specific cases, still have no precise solution, when considering the research done in this domain. Thus, an automated feature extraction process allows for better feature vector extraction from the data and get better classification accuracy.

An autoencoder is a neural network that efficiently applies the task of representation learning. It transforms data into a compressed knowledge and information representation, producing a feature vector that represents the information contained in a sample of the data. Additionally, as an autoencoder learns to compress the dimensionality of the data into a specific sized feature space, it learns how to ignore the noise in the data, thus allowing the use of the raw data with the minimal need for pre-processing steps.

CNNs are well known having the capability to extract features, and are popular specifically when working with image datasets as implementations of 2D CNNs. We use CNNs as an architecture for an autoencoder and apply 1D CNN implementations for the time-series data that is recorded from the sensors, allowing the extraction of a feature vector of defined dimensions automatically that represents a sample information in the defined vector dimensions. This information includes the 13 channels of the data, representing the four triaxial sensors, and the force sensor. To fit the data into a CNN, it is necessary for all the samples of the dataset to have the same number of timesteps, with a sample being defined as a letter recording. Given the different time of recording per letter, each sample is resampled into a defined number of timesteps equal to 64. This is chosen to allow the extraction of a feature space vector of size 256. This feature space dimension is assumed to be sufficient to allow for better classification. Fig. 6 shows the architecture of the autoencoder and the defined dimensions per layer of the network.

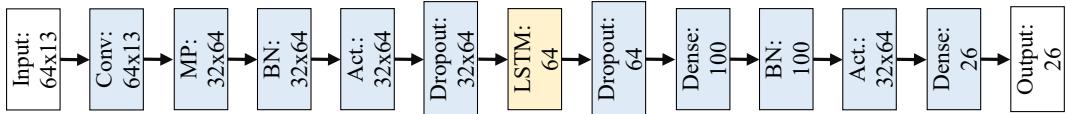


Fig. 7. Network structure of the CNN+LSTM. Layers: Conv = Convolution; Act. = Activation; BN = BatchNormalization; MP = MaxPooling1D.

4.1.4 ML-based Character Classification. Following the extraction of the specific features from the data, we use the complete feature vector as an input into several ML approaches to classify the written characters based on the features of the sensor data. We apply several classifiers using Python libraries. Online character classification is mainly based on techniques like kNN, HMM, SVM, LDA and NB.

As our baselines, we implemented Decision Tree, Random Forest, Logistic Regression, Linear SVM, and kNN. We perform grid search for the optimal hyperparameters that we line out in the following. **Decision Trees** (DTs) use tree-like structures of decisions and the possible consequences, in which each internal node represents a test on an attribute. The branch represents the evaluation of the test. The leaf node represents the class. We use DTs with default parameters, i.e., maximal depth and maximal leaf nodes are set to *None*. A **Random Forest** (RF) is an ensemble learning method that constructs a multitude of DTs while training. We apply a RF classifier with 100 trees, no defined max depth, a minimal sample split of 2, and minimal samples leaf of 1. For the **Logistic Regression** (LR) classifier we use a L2 norm for penalization, set the parameter $C = 1$, and set the tolerance for stopping criteria to 0.0001. We run the *lbfgs* solver maximal 100 iterations. **kNN** classifiers are non-parametric methods where the sample class is predicted by a plurality vote of its neighbors. If $k = 1$, the sample is assigned to the class of the single nearest neighbor. kNN is used in [17, 20, 36, 39] ($k > 1$) and in [60] ($k = 1$). We apply the kNN classifier considering five nearest neighbors ($k = 5$), we set the leaf size to 30 and weights to *uniform*. **Support Vector Machines** (SVM) classifiers are non-probabilistic classifier that is a representation of the samples as points in space, such that the samples of the separate categories are divided by a clear gap. For an SVM, also used in [17, 20, 32, 51, 60], a kernel with a gamma parameter and a slack variable has to be set. In our configuration, we use the slack variable $C = 1$, the tolerance 0.0001 and the L2 norm as penalty function, and trained maximal to a 1,000 iterations.

The stated classifiers are different methods that consider different attributes and approaches for applying classification over the available data features. As a result, these methods would produce different classification results and accuracies based on how the extracted features are functional for each classifier.

4.2 DL-based Character Classification

Classical classification approaches require the process of feature extraction from time-series data to train ML models. The feature extraction difficulty lies in the limitations of the expertise in that specific field. Autoencoders are established to be automated feature extraction methods, but in a two-stage training process, they are, however, not informed about the final classification task, and have hence no access to the complete information. Therefore, we present end-to-end DL methods that provide state-of-the-art results with no feature extraction. Similarly to fitting the data into the autoencoder, that data was resampled to have a fixed length of timesteps providing a form to fit the data into the different types of networks.

Liwicki et al. [46] used RNNs, i.e., a bidirectional Long-Short-Term Memory (BiLSTM), with the Connectionist Temporal Classification (CTC) [27] objective function for online whiteboard handwriting recognition (74.0 % accuracy), and showed an improvement over HMM-based systems (65.4 % accuracy) on the IAM-OnDB [45] dataset. LSTMs [31] are RNN architectures designed to bridge long time delays between relevant input and target events. BiLSTMs [64] are able to incorporate context on both sides of every position in the input sequence, e.g., in word

Table 4. Evaluation results. Accuracies are given in % for different classifier.

Method		Lowercase		Uppercase		Combined	
		WD	WI	WD	WI	WD	WI
ML-based Features	Random Forest	54.77	43.04	56.29	45.96	42.39	30.44
	Decision Tree	29.36	22.89	29.87	24.32	19.19	14.96
	Logistic Regression	55.36	49.60	58.54	53.26	43.95	39.11
	Linear SVM	61.55	51.07	63.70	54.00	48.77	38.71
	kNN	49.17	29.87	51.32	30.94	38.30	19.61
ML-based Autoencoder	Random Forest	58.02	45.55	63.19	43.73	43.60	43.62
	Decision Tree	30.49	21.73	33.23	19.68	20.32	20.33
	Logistic Regression	56.16	44.93	62.59	43.73	41.66	41.66
	Linear SVM	62.09	51.80	70.61	51.74	46.54	46.56
	kNN	42.43	34.09	57.49	36.68	33.08	33.08
DL- based	CNN	84.62	76.85	89.89	83.01	70.50	64.01
	LSTM	79.83	73.03	88.68	81.91	67.83	60.29
	CNN+LSTM	82.64	74.25	88.55	82.96	69.42	64.13
	BiLSTM	82.43	75.72	89.15	81.09	69.37	63.38

recognition where the information left and right of a given letter is useful. The RNN approach of [28] achieved 79.7 % accuracy on the online word recognition task. Dynamic and static neural networks are used in [1].

We implement the CNN, LSTM and BiLSTM networks with a similar design using different architectures. The design includes two layers of the architecture with a 40 % dropout rate, a fully connected layer with 100 units, followed by the output layer including the number of classes, 26 classes for either lowercase or uppercase letter classification, and 52 classes for the complete alphabet classification (see Fig. 7). The LSTM and BiLSTM hidden layers include 64 units each. For the CNN hidden layers, we use a configuration of 64 feature maps, and a kernel size of 4 with max pooling of size 2. We use a rectified linear unit activation in the hidden layers with the Softmax activation function in the output classification layer. The cross entropy loss function is applied with Adam optimization using a 0.001 learning rate.

5 RESULTS AND DISCUSSION

In this section we report results for both cases presented in Section 3.3, i.e., writer-dependent and writer-independent recognition, with the training and test dataset splits as shown in Table 2. Considering the WI case, the datasets are split based on writers, keeping the writers in the test dataset completely different from the ones in the training dataset, while in the WD case, a single writer could be included in both datasets. For the WI task, we present averaged results for a 5-fold cross validation. Table 4 shows the performance of the baseline classifiers described in Section 4. We see that (in most cases) classical ML models perform slightly better when they get presented feature vectors from the autoencoder model. While this is not at a significant level it still shows that hand-crafted engineering of features is unnecessary. Among all the ML models, the linear SVM performs best. However, yet the recognition rates of the SVM only reaches an accuracy of 71 % over the WD recognition. The other classical ML models, i.e., the DT and RF models, yield much lower results over the test dataset due to early overfitting of the models during the training process (they reached a 100 % recognition rate over the training dataset).

The best classification accuracy is obtained with the CNN model for almost all of the different cases. The CNN model reaches almost 85 % accuracy for the lowercase WD task, and 77 % accuracy for the lowercase WI task. The recognition rate increases to almost 90 % when classifying uppercase characters in the WD case, and to 83 % correct classification in the WI recognition. The results of the CNN+LSTM model and the BiLSTM model

92:16 • Ott and Wehbi, et al.

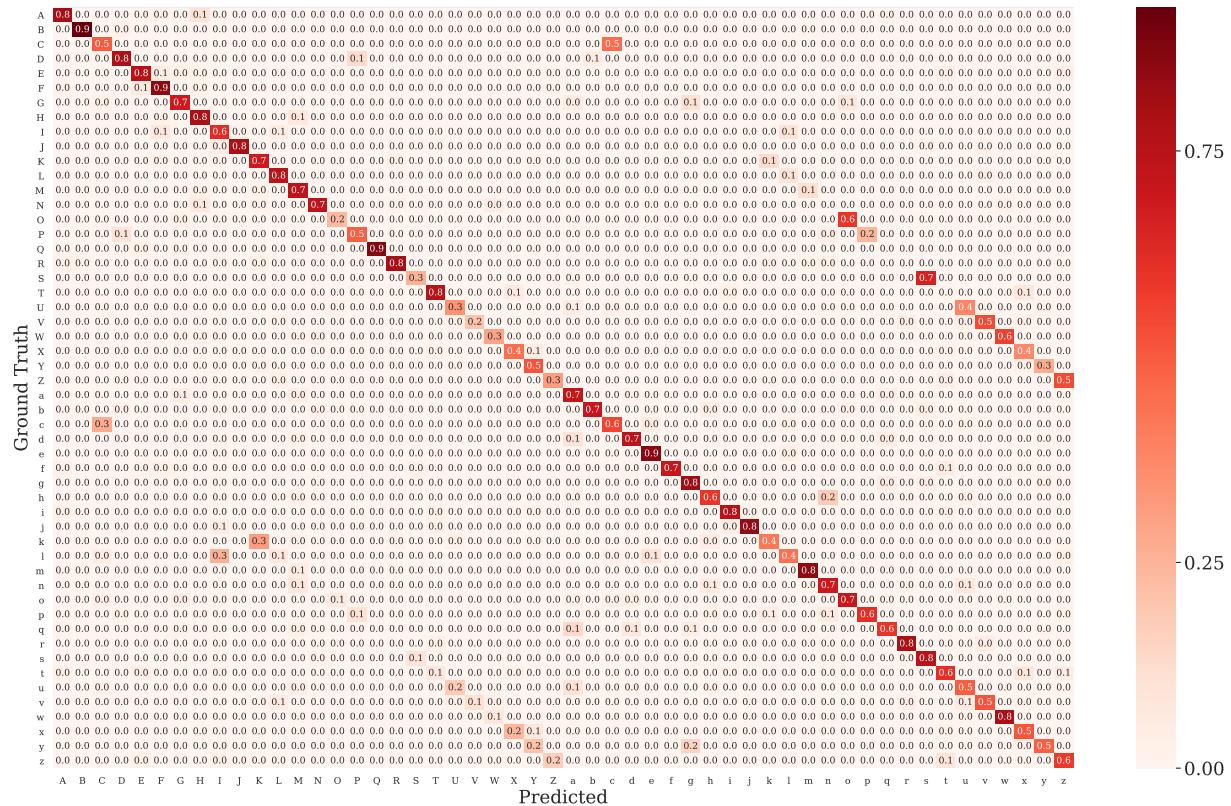


Fig. 8. Confusion matrix for predicted and ground truth WI combined letters. Presented are the CNN results.

are similar, with slight differences between the different cases. The LSTM provides the lowest accuracies when dealing the WI recognition.

The results show that state-of-the-art DL methods produce more accurate classification results than classical ML methods, even when considering an automated feature extraction method. We can also see that, in most cases, the best recognition rate is obtained at the uppercase letter classification. The accuracy drops for all cases when extending the classification into the complete 52 classes, as there are several characters that differ only in size and not in number of strokes, e.g., 'C'/'c', 'U'/'u', 'W'/'w', 'X'/'x', and 'Z'/'z', see the secondary diagonal of the confusion matrix in Fig. 8 for the CNN model for the WI combined case. The recognition rate is higher for the WD case in direct comparison to the WI case, showing that it is a more challenging task to accomplish as stated in Section 3.3.

The results presented can be improved with further investigation of the best hyperparameters in the models, and only serve as a baseline. When considering ML methods, better modeling of the data can be obtained using time-series analysis features that are not considered in our experiments, e.g., Wavelets and Shapelets. The dimensions of the feature vector, along with the autoencoder parameters can be improved for this classification task. Other DL models can be tested over the dataset with deeper hyperparameter optimization study to improve the recognition rate, specifically for the combined letter classification.

We evaluated the impact of each sensor on the results by training the models and leaving the data of one sensor out. The data of the magnetometer does not improve the classification accuracy. We publish the dataset including the magnetometer data for a possible investigation of further research.

The accuracies obtained with these experiments indicate that the presented dataset fulfills the requirements that are necessary for applying a writing recognition system stated in Section 2. The number of writers that contributed to the dataset allows a high recognition rate, and specifically grants the possibility of applying a recognition system that is able to recognize the handwriting of previously unseen users without having prior interaction or data. This makes the system a completely WI recognizer.

Since the number of contributions per writer to the dataset are approximately the same, the presented dataset shapes the 52 classes of the alphabet letters in balanced mode. This allows an implemented system to better distinguish between the available classes with less confusion between the letters to be recognized, specifically when dealing with similarly written characters. Although, the accuracy drops slightly for such characters, e.g., 'U' (75 % accuracy) and 'V' (69 % accuracy). The attribute similarity shown in Table 3 underlines that the average number of strokes are 1.01 for both letters, and the stroke deviation is small. Also the characters 'X' (66 % accuracy) and 'T' (82 % accuracy) are confused, as the recording attributes (X: $TS = 47.8$, $D_{TS} = 23.0$, $S = 1.81$, and $D_S = 0.64$, T: $TS = 47.5$, $D_{TS} = 19.7$, $S = 1.85$, and $D_S = 0.70$) are similar. Furthermore, placing no restrictions on the writers during the data recording sessions, such as writing speeds, directions and sizes, made the data as natural as possible. This allows the implemented systems to generalize the recognition to several different writing styles.

Additionally, using solely a sensor-enhanced pen to collect the data grants the possibility for the extension of the dataset, since no other devices are required in the process. Using the OnHW-chars dataset allows the implementation of a WI handwriting recognizer that only requires the use of a sensor-enhanced DigiPen.

6 CONCLUSION

In this paper, we addressed the handwriting recognition task and the available public datasets that are popular in the scientific community. We summarized available offline and online handwriting datasets, and made an in-depth comparison to our novel OnHW dataset that includes data for writing alphabet characters on regular paper. The dataset was collected using the STABILO Digipen. It consists of 31,275 letter samples, distributed into 15,650 lowercase and 15,625 uppercase letters collected from 119 writers who contributed approximately equally to the dataset. The dataset provides a time-series representation of sensor signals that recorded the pen movement during writing, which include linear accelerations, angular velocities and magnetic field recordings that help in identifying the angle at which the pen was held, along with the force applied by the pen on the paper to identify when writing and hovering occurs. To the extent of our knowledge, there are several attempts for applying online handwriting using sensor-enhanced pens, however no data used within these projects that covers character level recognition was made publicly available. This presented dataset forms a novel benchmark for future research to further improve online handwriting recognition, specifically character classification while writing on normal paper.

In addition, we implemented a series of experiments for the online character classification task, applied over different subsets of the dataset, based on multiple ML and DL algorithms, which are widely used in the time-series classification domain. We draw benefits of data pre-processing, feature extraction, and letter classification. The experimental results showed that CNNs achieve the best results when classifying characters over different subsets, achieving accuracies of 90 % for the WD and 83 % for the WI classification task on average. These presented models serve as benchmark models that can be used in the scientific community when applying character classification using sensor data provided from a pen.

ACKNOWLEDGMENTS

This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by the research program Human-Computer-Interaction through the project "Schreibtrainer" (grant number 16SV8228), and by the Bayerisches Staatsministerium für Wirtschaft, Landesentwicklung und Energie which is part of the EINNS project (Entwicklung Intelligenter Neuronaler Netze zur Schrifterkennung) (grant number IUK-1902-0004).

REFERENCES

- [1] Fevzi Alimoglu and Ethem Alpaydin. 1997. "Combining Multiple Representations and Classifiers for Pen-based Handwritten Digit Recognition". In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*, Vol. 2. Ulm, Germany, 637–640.
- [2] Abdullah Almaksour, Eric Anquetil, Solen Quiniou, and Mohamed Cheriet. 2010. "Personalizable Pen-Based Interface Using Lifelong Learning". In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Kolkata, India, 188–193.
- [3] María A. Pérez Alonso. 2015. "Metacognition and Sensorimotor Components Underlying the Process of Handwriting and Keyboarding and Their Impact on Learning. An Analysis from the Perspective of Embodied Psychology". In *Proc. Social and Behavioral Sciences*.
- [4] Christoph Amma, Dirk Gehrig, and Tanja Schultz. 2010. "Airwriting Recognition using Wearable Motion Sensors". In *Intl. Conf. on Augmented Human (AH)*. Megève, France.
- [5] Lisa Anthony and Jacob O. Wobbrock. 2010. "A Lightweight Multistroke Recognizer for User Interface Prototypes". In *Proc. of Graphics Interface (GI)*. 245–252.
- [6] Lisa Anthony and Jacob O. Wobbrock. 2012. "\$N\$-Protractor: A Fast and Accurate Multistroke Recognizer". In *Proc. of Graphics Interface (GI)*. 117–120.
- [7] Relja Arandjelović and Tevfik Metin Sezgin. 2011. "Sketch Recognition by Fusion of Temporal and Image-based Features". In *Journal of Pattern Recognition*. 1225–1234.
- [8] François Beuvens and Jean Vanderdonckt. 2012. "Designing Graphical User Interfaces Integrating Gestures". In *Intl. Conf. on Design of Communication*. Seattle, CA, 313–322.
- [9] François Beuvens and Jean Vanderdonckt. 2012. "UsiGesture: an Environment for Integrating Pen-based Interaction in User Interface Development". In *Intl. Conf. on Research Challenges in Information Science (RCIS)*. Valencia, Spain, 1–12.
- [10] François Beuvens and Jean Vanderdonckt. 2014. "UsiGesture: Test and Evaluation of an Environment for Integrating Gestures in User Interfaces". In *Intl. Journal of Human-Computer Interaction*, Vol. 7(2). 139–160.
- [11] François Bouteruche, Sébastien Macé, and Eric Anquetil. 2006. "Fuzzy Relative Positioning for On-Line Handwritten Stroke Analysis". In *Intl. Workshop on Frontiers in Handwriting Recognition (IWFHR)*. La Baule, France.
- [12] Martin Bresler, Daniel Prusa, and Václav Hlaváč. 2015. "Detection of Arrows in On-Line Sketched Diagrams Using Relative Stroke Positioning". In *Winter Conf. on Applications of Computer Vision (WACV)*. Waikoloa, HI, 610–617.
- [13] Jorge Calvo-Zaragoza and Jose Oncina. 2014. "Recognition of Pen-Based Music Notation: the HOMUS dataset". In *Intl. Conf. on Pattern Recognition (ICPR)*. 3038–3043.
- [14] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. "EMNIST: Extending MNIST to Handwritten Letters". In *Intl. Joint Conference on Neural Networks (IJCNN)*. Anchorage, AK, 2921–2926.
- [15] Dalbir and Sanjiv Kumar Singh. 2015. "Review of Online & Offline Character Recognition". In *Intl. Journal of Engineering and Computer Science (IJECS)*, Vol. 4(5). 11729–11732.
- [16] Adrien Delaye. [n.d.]. "Pen and Touch Datasets". <https://sites.google.com/site/adriendelaye/home/pen-and-touch-datasets>
- [17] Adrien Delaye and Eric Anquetil. 2013. "HBF49 Feature Set: A First Unified Baseline for Online Symbol Recognition". In *Intl. Conf. on Pattern Recognition (ICPR)*, Vol. 46(1). 117–130.
- [18] Adrien Delaye, Sébastien Macé, and Eric Anquetil. 2009. "Modeling Relative Positioning of Handwritten Patterns". In *Intl. Conf. on Graphonomics Society (IGS)*. Dijon, France.
- [19] Thomas Deselaers, Daniel Keysers, Jan Hosang, and Henry A. Rowley. 2015. "GyroPen: Gyroscopes for Pen-Input with Mobile Phones". In *Trans. on Human-Machine Systems*, Vol. 45(2). 263–271.
- [20] Mathias Eitz, James Hays, and Marc Alexa. 2012. "How Do Humans Sketch Objects". In *Trans. on Graphics*.
- [21] Maged M. Fahmy. 2010. "Online Signature Verification and Handwriting Classification". In *Journal on Ain Shams Engineering (ASEJ)*, Vol. 1(1). 59–70.
- [22] Nikos Fakotakis, Ergina Kavallieratou, and Kokkinakis George. 2002. "Handwritten Character Recognition based on Structural Characteristics". In *Intl. Conf. on Pattern Recognition (ICPR)*, Vol. 3.
- [23] Tobias Feigl, Sebastian Kram, Philipp Woller, Ramiz H. Siddiqui, Michael Philippsen, and Christopher Mutschler. 2020. RNN-Aided Human Velocity Estimation from a Single IMU. In *Journal of Sensors*, Vol. 20(13).
- [24] Tobias Feigl, Christopher Mutschler, and Michael Philippsen. 2018. Supervised Learning for Yaw Orientation Estimation. In *Intl. Conf. on Indoor Positioning and Indoor Navigation (IPIN)*. Nantes, France, 206–212.

[25] Vittorio Fuccella, Poika Isokoski, and Benoît Martin. 2013. "Gestures and Widgets: Performance in Text Editing on Multi-touch Capable Mobile Devices". In *Intl. Conf. on Human Factors in Computing Systems (CHI)*. 2785–2794.

[26] Sabrina Gerth, Annegret Klassert, Thomas Dolk, Michael Fliesser, Martin H. Fischer, Guido Nottbuschand, and Julia Festman. 2016. "Is Handwriting Performance Affected by the Writing Surface? Comparing Preschoolers', Second Graders', and Adults' Writing Performance on a Tablet vs. Paper". In *Journal on Frontiers in Psychology*, Vol. 7.

[27] Alex Graves, Santiago Fernández, Faustino John Gomez, and Jürgen Schmidhuber. 2006. "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks". In *Intl. Conf. on Machine Learning (ICML)*. 369–376.

[28] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. 2009. "A Novel Connectionist System for Unconstrained Handwriting Recognition". In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 31(5). 855–868.

[29] Patrick Grother. 1995. "NIST Special Database 19 Handprinted Forms and Characters Database". In *National Institute of Standards and Technology*.

[30] Isabelle Guyon, Lambert Schomaker, Rójean Plamondon, and Stanley A. Janet. 1994. "UNIPEN Project of On-line Data Exchange and Recognizer Benchmarks". In *Intl. Conf. on Pattern Recognition (ICPR)*, Vol. 2.

[31] Sepp Hochreiter and Jürgen Schmidhuber. 1997. "Long Short-Term Memory". In *Neural Computation (NC)*, Vol. 9(8).

[32] Heloise Hse and A. Richard Newton. 2004. "Sketched Symbol Recognition using Zernike Moments". In *Intl. Conf. on Pattern Recognition (ICPR)*, Vol. 1. Cambridge, UK, 367–370.

[33] Jonathan J. Hull. 1994. "Database for Handwritten Text Recognition Research". In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 16(5). 550–554.

[34] Eric Anquetil (UMR IRISA). 2020. "IMISKetchSDB". <https://www-intuidoc.irisa.fr/base-de-donnees-imisketchsdb/>

[35] Neelasagar K. and K. Suresh. 2015. "Real Time 3D-Handwritten Character and Gesture Recognition for Smartphone". In *Intl. Journal of Computer Applications (IJCA)*, Vol. 123(13). 1–8.

[36] Shaikh Jahidabegum K. 2015. "Character Recognition System for Text Entry Using Inertial Pen". In *Intl. Journal of Science, Engineering and Technology Research (IJSETR)*, Vol. 4.

[37] Birendra Keshari and Stephen M. Watt. 2008. "Online Mathematical Symbol Recognition using SVMs with Features from Functional Approximation".

[38] Minwoo Kim, Jaechan Cho, Seongjoo Lee, and Yunho Jung. 2019. "IMU Sensor-Based Hand Gesture Recognition for Human-Machine Interfaces". In *Journal of Sensors*, Vol. 19(18). 3827.

[39] Christopher Koellner, Marc Kurz, and Erik Sonnleitner. 2019. "What Did You Mean? An Evaluation of Online Character Recognition Approaches". In *Intl. Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Barcelona, Spain, 1–6.

[40] Joseph J. LaViola and Robert C. Zeleznik. 2007. "A Practical Approach for Writer-Dependent Symbol Recognition Using a Writer-Independent Symbol Recognizer". In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 29(11).

[41] Bo Li, Yijuan Lu, Afzal A. Godil, Tobias Schreck, Masaki Aono, Henry Johan, Jose M. Saavedra, and Shoki Tashiro. 2013. "SHREC'13 Track: Large Scale Sketch-based 3D Shape Retrieval". In *Eurographics Workshop on 3D Object Retrieval (3DOR)*. Girona, Spain.

[42] Bo Li, Yijuan Lu, Chen-Feng Li, Afzal A. Godil, Tobias Schreck, Masaki Aono, Martin Burtscher, Hongbo Fu, Takahiko Furuya, H. Johan, J. Liu, Ryutarou Ohbuchii, A. Tatsuma, and Changqing Zou. 2014. "SHREC'14 Track: Extended Large Scale Sketch-Based 3D Shape Retrieval". In *Eurographics Workshop on 3D Object Retrieval (3DOR)*. Strasbourg, France.

[43] PeiYu Li, Ney Renau-Ferrer, Eric Anquetil, and Eric Jamet. 2012. "Semi-Customizable Gestural Commands Approach and its Evaluation". In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Bari, Italy, 473–478.

[44] Yang Li. 2010. "Protractor: A Fast and Accurate Gesture Recognizer". In *Intl. Conf. on Human Factors in Computing Systems (CHI)*. 2169–2172.

[45] Marcus Liwicki and H. Bunke. 2005. "IAM-OnDB - An On-line English Sentence Database Acquired from Handwritten Text on a Whiteboard". In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Seoul, South Korea.

[46] Marcus Liwicki, Alex Graves, Horst Bunke, and Jürgen Schmidhuber. 2007. "A Novel Approach to On-Line Handwriting Recognition Based on Bidirectional Long Short-Term Memory Networks". In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*. 367–371.

[47] David Llorens, Federico Prat, Andrés Marzal, Juan Miguel Vilar, María Jóse Castro-Bleda, Juan Carlos Amengual, Sergio Barrachina Mir, Antonio Castellanos, Salvador Espa na Boquera, Jon Ander Gómez, Jorge Gorbea-Moya, Albert Gordo, Vicente Palazón-González, Guillermo Peris Ripollés, Rafael Ramos-Garijo, and Francisco Zamora-Martínez. 2008. "The UJIpenchars Database: a Pen-Based Database of Isolated Handwritten Characters". In *Intl. Conf. on Language Resources and Evaluation (LREC)*. Marrakech, Morocco.

[48] Leena Mahajan and G. A. Kulkarni. 2014. "Digital Pen for Handwritten Digit and Gesture Recognition Using Trajectory Recognition Algorithm Based on Triaxial Accelerometer - A Review". In *Intl. Journal of Innovative Research in Science, Engineering and Technology*, Vol. 3(4). 356–363.

[49] U.-V. Marti and H. Bunke. 2002. "The IAM-database: an English Sentence Database for Offline Handwriting Recognition". In *Intl. Journal on Document Analysis and Recognition (ICDAR)*. 39–46.

[50] C. S. Myers and L. R. Rabiner. 1981. "A Comparative Study of Several Dynamic Time-Warping Algorithms for Connected-Word Recognition". In *The Bell System Technical Journal*, Vol. 60(7). 1389–1409.

3. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning

92:20 • Ott and Wehbi, et al.

- [51] Ralph Niels, Don Willems, and Louis Vuurpij. 2009. "The NicIcon Database of Handwritten Icons for Crisis Management".
- [52] Michael Oltmans. 2007. "Envisioning Sketch Recognition: A Local Feature Based Approach to Recognizing Informal Sketches". In *PhD, Massachusetts Institute of Technology (MIT)*.
- [53] Tom Y. Ouyang and Randall Davis. 2009. "A Visual Approach to Sketched Symbol Recognition". In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*. 1463—1468.
- [54] Tse-Yu Pan, Chih-Hsuan Kuo, Hou-Tim Liu, and Min-Chun Hu. 2019. "Handwriting Trajectory Reconstruction Using Low-Cost IMU". In *Trans. on Emerging Topics in Computational Intelligence (TETCI)*, Vol. 3(3). 261–270.
- [55] Vietminh Paz-Villagrán, Jérémie Danna, and Jean-Luc Velay. 2013. "Lifts and Stops in Proficient and Dysgraphic Handwriting". In *Journal of Human Movement Science*, Vol. 33(1).
- [56] Réjean Plamondon and Sargur N. Srihari. 2000. "On-line and Off-line Handwriting Recognition: A Comprehensive Survey". In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 22(1). 63–84.
- [57] Anisha Priya, Surbhi Mishra, Saloni Raj, Sudarshan Mandal, and Sujoy Datta. 2016. "Online and Offline Character Recognition: A Survey". In *Intl. Conf. on Communication and Signal Processing (ICCSP)*. Melmaruvathur, India, 967–970.
- [58] Yosra Rekik, Radu-Daniel Vatavu, and Laurent Grisoni. 2014. "Match-up & Conquer: A Two-step Technique for Recognizing Unconstrained Bimanual and Multi-finger Touch Input". In *Intl. Working Conf. on Advanced Visual Interfaces (AVI)*. 201–208.
- [59] Yosra Rekik, Radu-Daniel Vatavu, and Laurent Grisoni. 2014. "Understanding Users' Perceived Difficulty of Multi-Touch Gesture Articulation". In *Intl. Conf. on Multimodal Interaction (ICMI)*. 232–239.
- [60] Ney Renau-Ferrer, Peiyu Li, Adrien Delaye, and Eric Anquetil. 2012. "The ILGDB Database of Realistic Pen-based Gestural Commands". In *Intl. Conf. on Pattern Recognition (ICPR)*. Tsukuba, Japan, 3741–3744.
- [61] Valérie Renaudin, Muhammad Haris Afzal, and Gérard Lachapelle. 2001. "Complete Triaxis Magnetometer Calibration in the Magnetic Domain". In *Journal of Sensors*.
- [62] J. M. Romeu, B. Lamiro, G. Sanchez, and J. Llados. 2006. "Automatic Adjacency Grammar Generation from User Drawn Sketches". In *Intl. Conf. on Pattern Recognition (ICPR)*. Hong Kong, China, 1026–1029.
- [63] Manju Sahu and Alpha Kujur. 2017. "Differentiation and Comparison of Left Handed and Right Handed Writers on the Basis of Strokes and Slope of Letter". In *Intl. Journal of Current Research and Review (IJCRR)*, Vol. 9(11). 6–9.
- [64] Mike Schuster and K. K. Paliwal. 1997. "Bidirectional Recurrent Neural Networks". In *Trans. on Signal Processing*, Vol. 45(11). 2673–2681.
- [65] Timothy J. Smoker, Carrie E. Murphy, and Alison K. Rockwell. 2009. "Comparing Memory for Handwriting versus Typing". In *Human Factors and Ergonomics Society (HFES)*.
- [66] STABILO DigiVision. 2020. "The STABILO Digipen: A Sensor-equipped Ballpoint Pen with Wireless Connectivity. The UbiComp 2020 Challenge". <https://stabilodigital.com/stabilo-digivision/>
- [67] C. C. Tappert. 1982. "Cursive Script Recognition by Elastic Matching". In *Journal of Research and Development (JRnD)*, Vol. 26(6). 765–771.
- [68] Caglar Tirkaz, Berrin Yanikoglu, and T. Metin Sezgin. 2012. "Sketched Symbol Recognition with Auto-completion". In *Journal of Pattern Recognition*. 3926–3937.
- [69] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. 2012. "Gestures as Point Clouds: A \$P Recognizer for User Interface Prototypes". In *Intl. Conf. on Multimodal Interaction (ICMI)*. 273–280.
- [70] Christian Viard-Gaudin, Pierre Michel Lallican, Stefan Knerr, and Philippe Binter. 1999. "The IRESTE On/Off (IRONOFF) Dual Handwriting Database". In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Bangalore, India, 455–458.
- [71] Alessandro Vinciarelli and Michael Peter Perrone. 2003. "Combining Online and Offline Handwriting Recognition". In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Edinburgh, UK, 844–848.
- [72] Jeen-Shing Wang and Fang-Chen Chuang. 2012. "An Accelerometer-Based Digital Pen with a Trajectory Recognition Algorithm for Handwritten Digit and Gesture Recognition". In *Trans. on Industrial Electronics (IES)*, Vol. 59(7). 2998–3007.
- [73] Jeen-Shing Wang, Yu-Liang Hsu, and Cheng-Ling Chu. 2013. "Online Handwriting Recognition Using an Accelerometer-Based Pen Device". In *Intl. Conf. on Computational Science and Engineering (CSE)*. Sydney, Australia, 229–232.
- [74] Kai Wang and Serge Belongie. [n.d.]. "Word Spotting in the Wild". In *Europ. Conf. on Computer Vision (ECCV)*. Heraklion, Crete.
- [75] R. Allen Wilkinson, Jon Geist, Stanley Janet, Patrick J. Grother, Christopher J. C. Burges, Robert Creecy, Bob Hammond, Jonathan J. Hull, Norman J. Larsen, Thomas P. Vogl, and Charles L. Wilson. [n.d.]. "The First Census Optical Character Recognition System Conference".
- [76] Don Willems, Ralph Niels, Marcel van Gerven, and Louis Vuurpijl. 2009. "Iconic and Multi-stroke Gesture Recognition". In *Journal of Pattern Recognition*, Vol. 42(12). 3303–3312.
- [77] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. 2007. "Gestures without Libraries, Toolkits or Training: A \$1 Recognizer for User Interface Prototypes". In *ACM Symp. on User Interface Software and Technology (UIST)*. 159–168.
- [78] Adeel Yousaf, Muhammad Jaleed Khan, M. Imran, and Khurram Khurshid. 2017. "Benchmark Dataset for Offline Handwritten Character Recognition". In *Intl. Conf. on Emerging Technologies (ICET)*. 1–5.
- [79] Ming Zeng, Le T. Nguyen, Bo Yu, Ole J. Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. 2014. "Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors". In *Intl. Conf. on Mobile Computing, Applications and Services (MobiCASE)*. Austin, TX, 197–205.

Chapter 4

Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

Contributing Article

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In *Proceedings of the IEEE/CVF Winter Conference for Applications on Computer Vision (WACV)*, pages 266–276, Waikoloa, HI, January 2022. doi:10.1109/WACV51458.2022.00131.

Publisher Website

<https://ieeexplore.ieee.org/document/9706891>

Copyright License

© 2022 IEEE. Reprinted, with permission, from Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In *Proceedings of the IEEE/CVF Winter Conference for Applications on Computer Vision (WACV)*, pages 266–276, Waikoloa, HI, January 2022. doi:10.1109/WACV51458.2022.00131.

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of Ludwig-Maximilians University Munich products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to <http://www.ieee.org>.

4. Joint Classification and Trajectory Regression of Online Handwriting using 134 a Multi-Task Learning Approach

[org/publications_standards/publications/rights/rights_link.html](http://www.rightslink.ca/) to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or Pro-Quest Library, or the Archives of Canada may supply single copies of the dissertation.

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing, visualization, and project administration. David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Lucas Heublein contributed for the methodology (i.e., creation of models), software, validation, investigation, and data curation (i.e., data recording and data management). Bernd Bischl contributed by supervision. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), visualization (i.e., presentation of the published work), supervision, and funding acquisition of the project “Schreibtrainer” (grant number 16SV8228) by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”. This paper was presented by Felix Ott as a poster and video at the IEEE/CVF Winter Conference for Applications on Computer Vision (WACV) in January 2022.

Datasets and Source Code

Along this paper, an online handwriting dataset was recorded and published. The source code to reproduce the evaluation results is publicly available at:

<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>

Statement about Recent, Related Research¹⁷

The proposed *Handwriting-Assistant* by Bu et al. (2021) captures free handwriting of ordinary pens with millimeter-level accuracy using an IMU attached to the pen’s back. A principal component analysis (PCA) method detects the candidate’s writing plane, and the distance variation of each segment relative to the plane is exploited to distinguish on-plane strokes. To evaluate the reconstructed trajectories, the authors used a dataset containing ground truth handwriting trajectories recorded on a touch screen surface. However, the average tracking error of the *Handwriting-Assistant* method is 1.84 mm evaluated on lower-case characters, which cannot be compared with the results of the contributing paper (Ott et al., 2022c) since it was evaluated on both lowercase (small size) and uppercase (larger size) characters. Additionally, the pen cap with the integrated IMU sensor is large, which

can influence the writing style, and hence, the pen may not be applicable in real-world scenarios.

Kuramoto et al. (2020) put forward the proposal for a six-axis force-torque sensor that can determine the contact state of a pen tip to calculate its position and trace. The method relies on kinematic computations of the pen tip and a probabilistic determination of the contact state. The authors test their method on written symbols, which is similar to the dataset proposed in the contributing paper (Ott et al., 2022c). Nonetheless, using such a device is necessary despite its disadvantage of potentially influencing the writing style.

The authors of ChiSig (Yan et al., 2022) proposed a pipeline for detecting, restoring, and verifying of Chinese signatures. To assess the restoration, they utilized metrics based on the Fréchet Inception distance and the similarity between corresponding pixels of images. In contrast, the reconstructed and ground truth trajectories in the contributing paper (Ott et al., 2022c) are represented as spatio-temporal time-series, and thus the Euclidean distance is used. Since there is no established metric for evaluating trajectory reconstructions, Chen et al. (2022b) propose two metrics: Adaptive interaction on union, which reduces the influence of stroke widths, and length-independent dynamic time warping, which solves the trajectory-point alignment problem. These metrics could be suitable for evaluating the reconstructed trajectories in the contributing paper (Ott et al., 2022c).

Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

Felix Ott^{1,2}

David Rügamer²

Lucas Heublein¹

Bernd Bischl²

Christopher Mutschler¹

¹Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits IIS

²LMU Munich, Munich, Germany

{felix.ott, christopher.mutschler}@iis.fraunhofer.de

{david.ruegamer, bernd.bischl}@stat.uni-muenchen.de

Abstract

Multivariate Time Series (MTS) classification is important in various applications such as signature verification, person identification, and motion recognition. In deep learning these classification tasks are usually learned using the cross-entropy loss. A related yet different task is predicting trajectories observed as MTS. Important use cases include handwriting reconstruction, shape analysis, and human pose estimation. The goal is to align an arbitrary dimensional time series with its ground truth as accurately as possible while reducing the error in the prediction with a distance loss and the variance with a similarity loss. Although learning both losses with Multi-Task Learning (MTL) helps to improve trajectory alignment, learning often remains difficult as both tasks are contradictory. We propose a novel neural network architecture for MTL that notably improves the MTS classification and trajectory regression performance in online handwriting (OnHW) recognition. We achieve this by jointly learning the cross-entropy loss in combination with distance and similarity losses. On an OnHW task of handwritten characters with multivariate inertial and visual data inputs we are able to achieve crucial improvements (lower error with less variance) of trajectory prediction while still improving the character classification accuracy in comparison to models trained on the individual tasks.

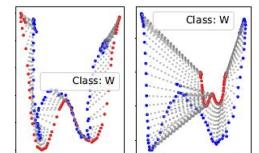
1. Introduction

MTS Classification (MTSC) identifies labels based on MTS as a set of real-valued, sequentially ordered observations. MTSC appears across many domains, e.g., human activity recognition (HAR), handwriting reconstruction, and medical data analysis. [31, 46, 60, 62] A related yet different task is trajectory reconstruction and function alignment.

This is important to applications that involve the modeling of mathematical functions or for shape analysis, e.g., to optimally transform a shape into another shape [19, 53].

An application where both tasks must be solved simultaneously is HAR [1, 4]. For example, the control of smart devices through hand-movement patterns or sport applications [35] requires a joint learning of the pattern classification and the hand trajectory. The data are recorded using a handheld device with inertial sensors or by outside-in cameras, e.g., a Kinect system. For our handwriting recognition application, common techniques require to write on a device where the writing style is influenced, to take images of the handwritten text, or to use a stylus pen, a touch pen with a sensible magnetic mesh tip together with a touch screen surface [2]. Systems for writing on paper are only prototypical, such as the ones used in [12, 50, 58], or are smartphones that provide a pen-like interaction from standard built-in sensors [17]. For our OnHW application, we used a novel sensor-enhanced pen [33, 46] and recorded the pen movement with outside-in cameras.

Combined metrics. In computer vision tasks such as landmark localization [9] and human pose estimation [41], DL has successfully contributed to sequence to sequence regression-based methods [15]. However, in trajectory prediction we do not only want to align reconstructed trajectories with their ground truth, but also require



(a) Distance loss. (b) Similarity loss.
Figure 1: Ground truth (red) and reconstructed (blue) trajectories.

them to be smooth over time [59]. A combination of distance metrics with geometric shape-based and spatio-temporal metrics achieves such a smoothness [34]. However, both metrics contradict each other which makes them difficult to be used together for training (cf. the blue trajec-

Input	Task	Loss	Output
Inertial or visual MTS	Classification Regression	Cross-entropy Distance Spatio-temporal Distribution	Character class or/and trajectory (MTS)

Table 1: Summary of the challenge addressed in this paper.

tory of the handwriting reconstruction task in Fig. 1. Optimizing a *distance* loss (here: Euclidean, Fig. 1a) introduces a relative error and dilatation while a *spatio-temporal* loss (here: Pearson Correlation, Fig. 1b) provides a smooth and similarly shaped trajectory, but with (large) scaling error.

We address the problem of learning both metrics simultaneously by MTL. MTL exploits differences and commonalities across two or more single learning tasks to solve them jointly with possibly improved performance in each single task. As the usefulness of different tasks is not known a priori, combining loss functions is one of the main challenges. The goal in MTL is to find appropriate weighting strategies such that the total loss is minimized optimally [13, 20, 40]. MTL research made significant progress over the last years regarding training techniques [44, 61], but is still challenging to apply for such contradictory tasks.

In this paper we propose different MTL architectures that combine two heterogeneous but subtly correlated tasks: MTSC and trajectory regression on two different datasets from an OnHW recognition application [46]. Table 1 summarizes the challenge we address. We use MTL [40] and show that the order of weight increase is crucial for smooth trajectory regression. Our results show an improved character classification and optimal trajectory regression by combining the cross-entropy loss with distance and similarity losses, i.e., spatio-temporal metrics.

The paper is organized as follows. We discuss related work in Sec. 2. Sec. 3 explains how we combine classification and regression loss functions in an MTL setup. Sec. 4 presents our novel OnHW datasets and CNN architectures before Sec. 5 shows experimental results. Sec. 6 concludes.

2. Related Work

We discuss related work on MTSC (Sec. 2.1), loss functions for trajectory regression (Sec. 2.2), and MTL (Sec. 2.3).

2.1. Deep Learning based Multivariate Time Series Classification (MTSC)

MTSC is used in different fields to estimate class labels based on several (in-)dependent time series. While there are many shapelet- and Fourier-based methods, we focus on DL methods as they are most similar to our approach. DL-based methods often exploit LSTM and CNN layers to extract features. Examples include the multi-channel CNN for univariate processing by [62] and the attention-based LSTM by [29]. [31] introduced a squeeze-and-excitation block to

generate latent features for classification (MLSTM-FCN). [60] proposed the attentional prototype TapNet that handles the issue of limited training labels combined with a random group permutation method. An overview of MTSC methods can be found in [22]. However, as such approaches focus on classification only they perform poorly on (smooth) trajectory regression.

2.2. Trajectory Regression

For the reconstruction of time series we need a metric to measure the similarity between a predicted time series and its ground truth. We here briefly review known metrics and discuss them in the context of reconstructing trajectories.

Commonly used *distance*-based metrics are the Mean Squared Error (MSE), the Mean Absolute Error (MAE) (being more robust to outliers but with potentially large gradients near the optimum), the Huber loss [30], Andrew’s Sine, and Tuckey’s Biweight [7], each handling “outliers” differently [8]. Although practically relevant, these methods do not consider temporal dimensions or guarantee smoothness.

Geometric shape-based similarity measures are the Fréchet distance [10], preserving the time series order of sequence data along curves, the Hausdorff distance [55], a measure for dissimilarity for comparing point sets, and the Procrustes analysis [51]. [59] proposed a trajectory simplification by using sub-trajectory similarity information by the Fréchet and Hausdorff distances. However, optimizing such metrics is often difficult as they are not differentiable.

Spatio-temporal distance metrics take into account both the spatial and the temporal dimensions of movement data, such as time-dependent trajectories. Examples include the Cosine Similarity and Pearson Correlation [47]. However, these metrics are inappropriate for shape reconstruction being invariant to scaling and translation.

Time Warping approaches such as Dynamic Time Warping (DTW) [11, 14] can compare time series of variable size and are robust to shifts or dilatation across time. Approaches typically solve a minimal-cost alignment problem solved with dynamic programming [15] using the MSE or the Mahalanobis distance [24]. For audio-to-audio alignment [24] proposed to learn Hamming and Mahalanobis metrics. [17] used dynamic programming to compare reconstructed trajectories, but only for result evaluation. However, such methods may lead to pathological results as warping the x-axis can produce unintuitive alignments.

Distribution-based methods exploit the distributional discrepancy of samples. One important example is the Kullback-Leibler (KL) divergence [3]. Optimal Transport compares probability distributions in a geometrically faithful way, but is limited because of its computational burden (e.g., the Wasserstein distance [23]). [28] reconstructed sketches using RNNs that are, however, represented as time-independent vector images and not trajectories. As the KL

4. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

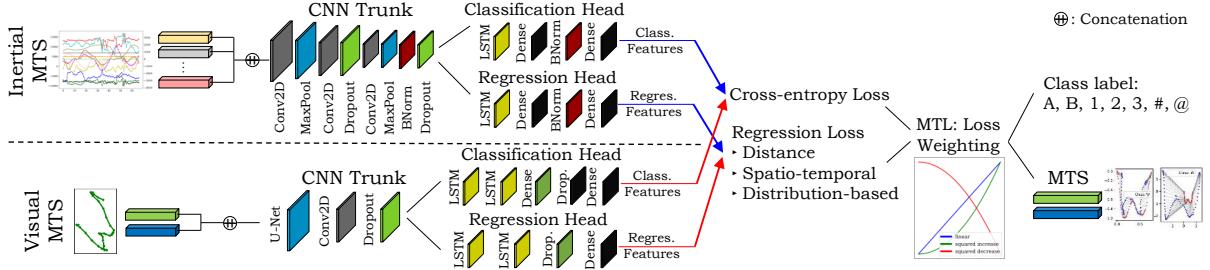


Figure 2: Method overview: MTS input, exemplary CNN trunk and heads, and class and trajectory prediction. The top row shows the architecture for the IMU (inertial measurement unit) dataset, the bottom row shows the architecture for the visual dataset. Classification (cross-entropy) and regression (distance, spatio-temporal and distribution-based) loss functions are combined with different MTL weighting strategies. \oplus : concatenation. The right part of the pipeline is equal for both datasets and CNNs, but the input is separate (inertial features or visual features). Different types of data are not combined.

divergence is asymmetric and does not satisfy the triangle equality, we will focus on the Wasserstein metric.

Various approaches for handwriting regression have been proposed, such as the application of adversarial domain adaptation for training generative RNNs on handwriting generation with MNIST [26], or sequence order inference by combining the L_2 norm and the Pearson Correlation [34]. However, they are training on relative distance pairs.

2.3. Multi-Task Learning (MTL)

MTL achieves a provable information gain over single task learning if the jointly learned tasks are somehow related [16]. A naive approach combines multiple losses using a weighted sum of losses of the single tasks. However, the model performance is very sensitive to the actual weight selection. Hence, [32] considered the homoscedastic uncertainty of each task to weight multiple tasks differently. [61] addressed the problem of learning heterogeneous but subtly correlated tasks (that have different convergence rates and learning difficulties) with task-wise early stopping and task-constrained models. Further different weighting strategies of the combination of single tasks exist, i.e., Dynamic Weight Average (DWA) [40] and Dynamic Task Prioritization [27], that dynamically prioritize difficult tasks during training. The methods by [13] (GradNorm) and [20] (GCS) are based on gradients for loss scaling, while [21] adds connections between layers, and hence, these methods depend on the network architecture. [44] addresses the challenge of combining auxiliary tasks into a single coherent loss by learning (non-)linear interactions between auxiliary tasks.

3. Methodology

We now present the problem formulation and methodological foundation of our approach. Fig. 2 gives an overview of our method for both the inertial and visual datasets. We encode the input data sources with a CNN trunk and process the features for each individual task with

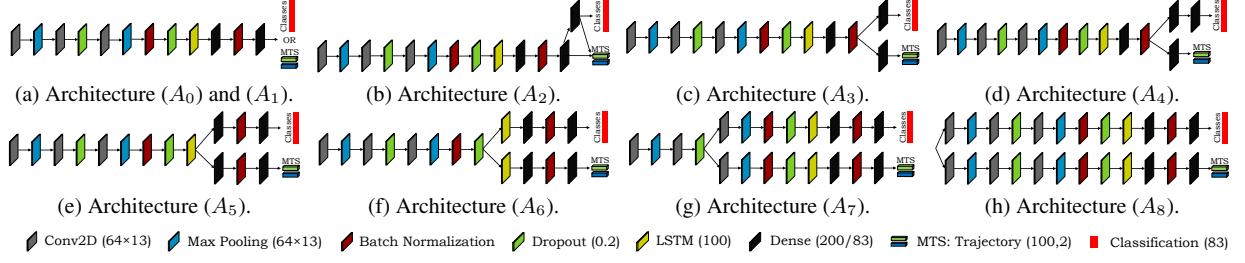
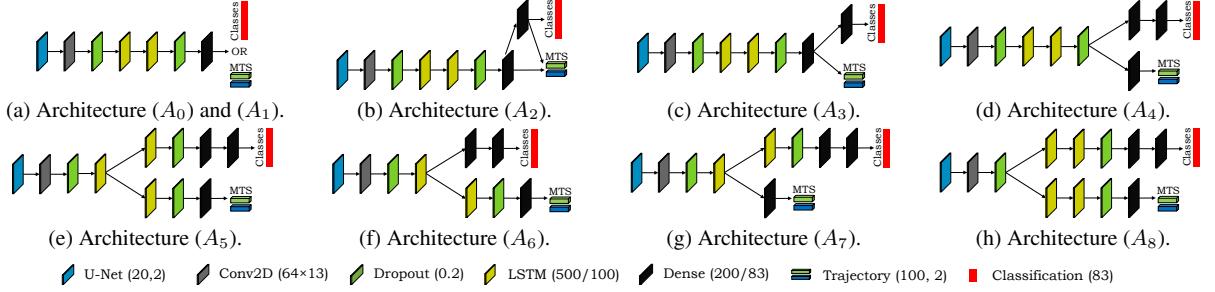
separated heads. We will first describe details for the MTSC task (i.e., following the classification head) in Sec. 3.1. For the trajectory prediction task (i.e., along the regression head) we make use of *distance*, *spatio-temporal* and *distribution-based* loss functions introduced in Sec. 3.2. We then present different MTL strategies for the combination of loss functions in Sec. 3.3, and propose suitable MTL architectures in Sec. 3.4 that allow to predict the class labels and MTS trajectories. Details are given in the Appendix A.1.

3.1. Multivariate Time Series (MTS) Classification

An MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $m \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l})$, $i \in \{1, \dots, m\}$, where m is the length of the time series and l is the number of dimensions. For example in pose tracking, we might have several streams induced by sensors attached to the body plus (features extracted from) a video stream. Each MTS is associated with a class label $v \in \Omega$ from a pre-defined label set Ω . The training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_n\} \in \mathbb{R}^{n \times m \times l}$, where n is the number of time series, and the corresponding labels $\mathcal{V} = \{v_1, \dots, v_n\} \in \Omega^n$ [60]. The MTSC task is to predict an unknown class label \hat{v} for a given MTS. We learn the classification model using the cross-entropy loss $\mathcal{L}_{CE}(\mathcal{U}, \mathcal{V})$ [25]. For details, see Appendix A.1.

3.2. Trajectory Regression Metrics

When reconstructing or regressing trajectories, such as handwritten characters, we have to consider another multi-dimensional (discrete) time series of varying length. The MTS can take values in $\Psi \subset \mathbb{R}$ and is represented by a matrix of size $n \times d$ (n : number of timesteps, d : dimensions of the time series). Given a ground truth time series $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_m\} \in \mathbb{R}^{m \times d}$, the goal is to predict a time series $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d}$, such that \mathcal{X} is closely aligned to \mathcal{Y} [34]. For our OnHW recognition task, the prediction is of size $(100, 2)$, but can be chosen arbitrarily for different applications. In the following, we consider

Figure 3: Overview of **inertial**-based architectures. STL: (A_0) and (A_1) . MTL (A_2) to (A_8) .Figure 4: Overview of **visual**-based architectures. STL: (A_0) and (A_1) . MTL (A_2) to (A_8) .

\mathcal{X} and \mathcal{Y} to be of same length n , and $\mathbf{r}_i = \mathbf{y}_i - \mathbf{x}_i$ be the residual between \mathcal{X}_i and \mathcal{Y}_i . We consider a (differentiable) substitution-cost function $\mathcal{L} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$ to learn the trajectory regression task. Different loss functions have different challenges and advantages. We make use of *distance-based*, *spatio-temporal* and *distribution-based* loss functions. For more details, see Appendix A.1.

As our **distance**-based loss function we consider the mean squared error $\mathcal{L}_{MSE}(\mathcal{X}, \mathcal{Y}) = \frac{1}{n} \|\mathcal{X} - \mathcal{Y}\|_2^2 = \frac{1}{n} \sum_{i=1}^n \mathbf{r}_i^2$ with L_2 -norm $\|\cdot\|_2$, the Huber loss $\mathcal{L}_H(\mathcal{X}, \mathcal{Y}, \delta_H)$ [30], which is less sensitive to outliers but depends on a hyperparameter δ_H , and the Andrew's Sine loss $\mathcal{L}_{AS}(\mathcal{X}, \mathcal{Y}, \delta_{AS})$ [8] with hyperparameter δ_{AS} . *Distance*-based loss functions, however, do not consider relative differences in input pairs.

The following **spatio-temporal** loss functions take into account the temporal dimensions of the data and maximize the shape similarity between ground truth and predicted trajectory. The Cosine Similarity is a measure of similarity between two non-zero vectors of an inner product space. The loss is defined by $\mathcal{L}_{CS}(\mathcal{X}, \mathcal{Y}) = 1 - (\mathbf{x} \cdot \mathbf{y}) / (\|\mathbf{x}\|_2 \|\mathbf{y}\|_2)$. Cosine Similarity is not invariant to shifts that is required for our application. The Pearson Correlation [47], in contrast, is invariant to shifts as it measures the linear relationship between two distributions in $[-1, 1]$, with 1 being a perfect alignment. Instead of the symmetric distance prediction [34], we train our model based on the Pearson Correlation loss $\mathcal{L}_{PC}(\mathcal{X}, \mathcal{Y}) = \mathcal{L}_{CS}(\mathcal{X} - \bar{\mathcal{X}}, \mathcal{Y} - \bar{\mathcal{Y}})$ with $\bar{\mathcal{X}}$ and $\bar{\mathcal{Y}}$ the mean of \mathcal{X} and \mathcal{Y} , respectively.

Finally, we also consider **distribution**-based loss func-

tions. Specifically, we use the Wasserstein distance [23] that defines a distance between two probability distributions on a given metric space M and that represents the cost δ of an optimal mass transportation problem. To solve the learning problem, we need to minimize the loss $\mathcal{L}_{WAS_p}(\mathcal{X}, \mathcal{Y})$, but calculating the gradient is computationally expensive. Hence, we optimize a smoothed Wasserstein loss function that is strictly convex [18].

3.3. Multi-Task Learning (MTL)

Our goal is to jointly classify an MTS using the cross-entropy loss and regress the corresponding trajectory, see the right part in Fig. 2. For each task, we have separate architecture heads. We show that both tasks are related, and hence, the MTL approach takes advantage of the information gain over single task approaches. The training of different tasks is non-trivial (see, [38, 39, 42, 61]). We have a set of tasks $K = \{T_1, \dots, T_{|K|}\}$ with $|K|$ tasks. The naive approach is to combine losses by a weighted sum $\mathcal{L}_{total} = \sum_{i=1}^{|K|} \omega_i \mathcal{L}_i$, with pre-specified, constant weights ω_i . We use this technique as a baseline, and choose $\omega_i = 1, \forall i \in \{1, \dots, |K|\}$ as default value, i.e., we weight the regression and classification losses equally. For trajectory regression we additionally combine two losses, namely *distance*-based metrics such as the MSE \mathcal{L}_{MSE} , Andrew's Sine \mathcal{L}_{AS} or Huber \mathcal{L}_H loss, with *spatio-temporal* distances such as the Cosine Similarity \mathcal{L}_{CS} or the Pearson Correlation \mathcal{L}_{PC} , or *distribution*-based loss functions, i.e., the Wasserstein metric \mathcal{L}_{WAS_p} . We perform different weighting strategies for these losses. First, we apply the naive

4. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

approach. Our second approach is to perform an epoch-dependent weighting where the weighting of the second task is dependent on the training process, i.e., the epoch number. We apply linear weight increase, squared weight increase, and squared weight decrease with respect to the weight of the second regression loss (see the blue, green and red lines in Fig. 2). As a third option, we apply DWA [40] by averaging task weighting over time. In detail, we define the weights for the current epoch e as

$$\omega_i(e) = \frac{e^{\lambda_i(e-1)/P}}{\sum_k e^{\lambda_k(e-1)/P}}; \quad \lambda_i(e-1) = \frac{\mathcal{L}_i(e-1)}{\mathcal{L}_i(e-2)}, \quad (1)$$

where P is a pre-specified softness of task weighting. For large P , $\lambda_i \approx 1$, and tasks are weighted equally. We set $P = 1$. λ_i is the relative descending rate between previous epochs $e-1$ and $e-2$ and is in the range $(0, +\infty)$.

3.4. MTL-specific Architectures

We consider different architectures to jointly learn the two tasks. For the MTL approach, lower representation layers (trunk) have to be shared between different tasks by forking into task-specific separate layers (heads). The ratio between trunk and heads is particularly important for our application. We train the following nine architectures: only regression (A_0), only classification (A_1), and MTL-based architectures (A_2 to A_8) with different split points (from A_2 being the latest split to A_8 being the earliest split). For our experiments we implement two different feature encoders: (1) a CNN that extracts features from the channels of the IMU (see Fig. 3), and (2) a CNN that extracts features from the visual dataset (see Fig. 4). The output of the CNNs are either the class for the MTSC task, the trajectory for the regression task, or both for the MTL approach. For details, see Appendix, Sec. A.2.

4. Joint Classification and Trajectory Regression of Online Handwriting (OnHW)

There exists no state-of-the-art dataset that contains ground truth trajectories and classification labels. We recorded two novel datasets for OnHW recognition of characters with a sensor-enhanced pen written on a tablet for ground truth and three outside-in cameras for pen tip reconstruction.

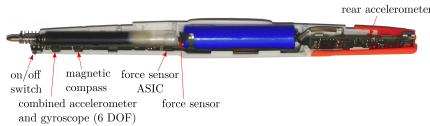
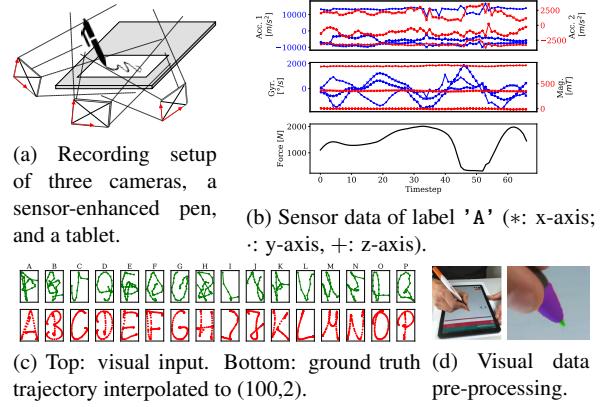


Figure 5: Pen with an integrated gyroscope, magnetometer, front and rear accelerometers, and a force sensor.



(c) Top: visual input. Bottom: ground truth (d) Visual data trajectory interpolated to (100,2). pre-processing.

Figure 6: Data recording of our *OnHW* dataset.

Recording Setup. Our inertial dataset uses a sensor-enhanced pen [46] that contains two accelerometers (3 axes each), one gyroscope (3 axes), one magnetometer (3 axes), and one force sensor at 100 Hz (Fig. 5). We replace the pencil lead with a Wacom EMR module and record ground truth trajectories at 30 Hz on a Samsung Galaxy Tab S4 tablet. Our visual dataset uses three cameras pointing on the pen (Fig. 6a) to record the movement of the pen tip at 60 Hz . A right-handed person wrote $\approx 18 \times 83$ characters containing small (26) and capital (26) letters, numbers (10), and symbols (21).

Pre-processing and Dataset. Fig. 6b shows an IMU signal of the letter 'A' (from the inertial dataset) and Fig. 6c shows characters 'A' to 'P' (from the visual dataset) from the pre-computed trajectory in camera coordinates (top row) and the ground truth trajectory produced by the tablet (bottom row). For the camera-based trajectories we segment the pixels of 100 random images of the dataset in the classes "pen" (the purple parts of the pen in Fig. 6d), "pen tip" (green parts of the pen), and "background". We train U-Net [49] to predict the pen tip from all images and choose the middle 90th percentile of 20 top-left pen tip pixels as the trajectory in camera coordinates (Fig. 6c). We interpolate the ground truth trajectory to (100,2). A 71/29 train/validation split results in 822 training and 332 validation characters for the IMU dataset, and in 2,466 training and 992 validation characters for the visual dataset. Datasets and source code publicly available upon publication.¹

CNN Architectures. The visual time series input is of size $m = 40$ due to 20 pixels in both camera axes, the length l is variable and dependent on the length of the character. For the IMU input the time series is of size $m = 13$ (for the two accelerometers, the gyroscope and the magnetometer

¹Dataset and source code: <https://iis.fraunhofer.de/onhw-dataset/>

Network	MSE+CE Traj. Class.	AS+CE Traj. Class.	H+CE Traj. Class.	MSE+PC+CE Traj. Class.	MSE+CS+CE Traj. Class.	MSE+WAS+CE Traj. Class.	PC+CE Class.	CS+CE Class.	WAS+CE Class.
Only regression (A_0)	<u>0.1705</u>	-	0.1594	-	0.1501	-	0.1723	-	1.0023
Only classification (A_1)	-	88.11	-	-	-	-	-	-	-
Class. for regr. (A_2)	0.1169	86.69	0.1779	9.78	0.1290	62.78	0.1127	86.81	0.3554
Latest split (A_3)	0.1381	86.67	0.1856	49.31	0.1569	66.73	0.1381	85.75	6.1464
Late split (A_4)	0.1372	86.46	0.1421	76.28	0.1581	63.64	0.1357	88.64	1.3928
Split after LSTM (A_5)	0.1370	87.34	0.1629	68.64	0.1458	73.74	0.1386	85.53	1.0578
Split after 2. Drop. (A_6)	0.1623	87.68	0.1464	83.96	0.1580	84.76	0.1647	84.94	1.0053
Split after 1. Drop. (A_7)	0.1866	84.27	0.1676	86.93	0.1546	86.89	0.1638	84.55	1.1388
Separate heads (A_8)	0.1936	86.87	0.1660	86.02	0.1533	88.43	0.1490	87.03	1.0986
									0.3315
									82.03
									87.15
									88.15
									82.58

Table 2: Evaluation results for the IMU-based dataset trained with different loss combinations. Trajectory evaluation metric: root mean squared error (RMSE). Classification accuracy given in %. A_0 and A_1 : single task architectures. A_2 to A_8 : MTL architectures. Underlined: baselines. **Bold**: best results. Columns are combinations of different loss functions, e.g., MSE+CE is a combination of the \mathcal{L}_{MSE} and the \mathcal{L}_{CE} losses, etc.

Network	MSE+CE Traj. Class.	AS+CE Traj. Class.	H+CE Traj. Class.	MSE+PC+CE Traj. Class.	MSE+CS+CE Traj. Class.	MSE+WAS+CE Traj. Class.	PC+CE Class.	CS+CE Class.	WAS+CE Class.
Only regression (A_0)	0.1360	-	0.1388	-	0.1271	-	0.1348	-	2.5733
Only classification (A_1)	-	<u>73.19</u>	-	-	-	-	-	-	-
Class. for regr. (A_2)	0.1250	80.15	0.1279	12.47	0.1475	55.81	0.1327	77.27	0.4900
Latest split (A_3)	0.4314	23.21	0.1327	74.3	0.1577	64.15	0.1401	75.54	0.6713
Late split (A_4)	0.1351	78.49	0.1260	78.23	0.1230	79.42	0.1284	80.89	1.1177
Split after LSTM (A_5)	0.1291	80.40	0.1252	81.24	0.1307	78.06	0.1478	73.21	0.1705
LSTM in tr. head (A_6)	0.1334	77.33	0.1383	74.31	0.1605	63.31	0.1243	81.64	0.1747
LSTM in cl. head (A_7)	0.1378	78.58	0.1267	80.52	0.1330	78.86	0.1459	74.25	1.2483
Split after 1. Drop. (A_8)	0.1246	78.56	0.1248	75.69	0.1310	75.18	0.1251	79.27	0.4132
									0.2448
									75.83
									80.76
									79.42

Table 3: Evaluation results for the visual dataset trained with different loss combinations. For definitions, see Table 2.

with 3 axes each plus the force sensor) with a variable length l . When training both encoders, each batch is bias shuffled, such that each batch contains letters of approximately the same time step length. To account for variable batch length, we use zero padding to the maximal size in each batch. For the classification task the last *dense* layer has 83 neurons, for the trajectory regression task 200 neurons (reshaped: 100×2). In a preliminary study, we searched for the optimal dropout and LSTM neurons for the visual CNN architecture (see Appendix, Fig. 11). We choose an LSTM combination of 500 and 100 units, and two dropout layers with 20% dropout after the convolution and the second LSTM layer.

5. Experimental Results

We here describe the results for the IMU dataset (Sec. 5.1 to 5.3, Table 2) and for the visual dataset (Sec. 5.4, Table 3). More specifically, Sec. 5.1 compares the proposed MTL architectures against the baseline networks A_0 and A_1 . We use the cross-entropy loss for MTSC, and *distance*, *spatio-temporal*, and *distribution*-based losses for trajectory prediction (Sec. 5.2). In Sec. 5.3, we evaluate our MTL strategies, and compare our methods to state-of-the-art techniques in Sec. 5.5.

Preliminary. We first want to emphasize that improvements in the smoothness of the predicted trajectory are utmost importance in our handwriting application, but that smoother trajectories do not necessarily lead to a significant improvement of the reconstruction error. This is similar to image reconstruction, where the biggest gain in performance is achieved by vaguely reconstructing the image, yet the image still looks unnatural for humans. Hence, also small trajectory improvements are of interest.

Hardware and Training Setup. For all experiments we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the vanilla Adam optimizer with learning rate 10^{-4} . We run each experiment three times for 20,000 epochs with a batch size of 50 and report averaged results (over five epochs).

Evaluation Metric. For evaluation we compute the categorical metric for class prediction in %, and the root mean squared error (RMSE) for trajectory prediction. As in [59], we also compute the *geometric shape*-based Fréchet [10], Hausdorff [55] and Procrustes [51] measures, as well as the *time warping* approach DTW [15] (see Sec. 2.2). Due to a correlation between these metrics with the RMSE and for better readability, we only report the RMSE.

4. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

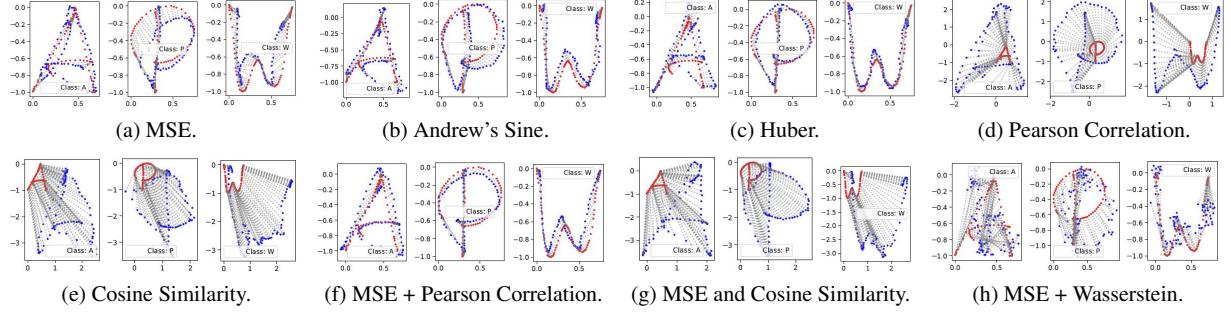


Figure 7: Trajectory prediction (blue) against the ground truth trajectory (red) of the characters 'A', 'P' and 'W' based on inertial data trained with different combinations of loss functions.

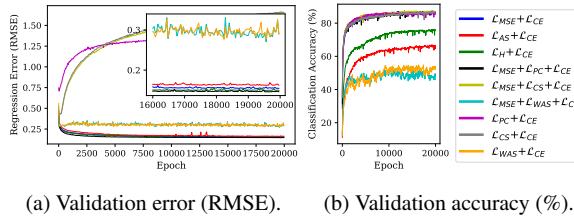


Figure 8: Evaluation of combinations of loss functions averaged over all architectures (A_0 - A_8). MTL strategy: naive weighting $w = 1$. Baseline (blue): $\mathcal{L}_2 + \mathcal{L}_{CE}$.

5.1. MTL Architecture Evaluation

As a baseline for the inertial dataset, model A_0 results in an error of 0.1705 using the \mathcal{L}_{MSE} , and model A_1 in an accuracy of 88.11%. Exemplary reconstructed letters of the baseline method is shown in Fig. 7a. The trajectory regression task improves up to 0.1169 for model A_2 to 0.1623 for model A_6 , but decreases for model A_7 and A_8 . The accuracy of 86.69% (A_2) is less accurate than the baseline A_1 (87.68%). We conclude, that a late split has a positive influence on the trajectory regression by sharing more trainable parameters in the trunk. This even holds for smaller model sizes (see Appendix, Table 5). For a larger regression head (A_7 and A_8) the model is prone to overfitting.

5.2. Loss Function Evaluation

Single Loss Functions. For more details on hyperparameter searches, see Appendix Sec. A.3. For the *distance*-based metrics, i.e., Andrew's Sine and Huber we observe a decrease in trajectory error (Fig. 7b and 7c) compared to \mathcal{L}_{MSE} at the cost of a deterioration of the classification accuracy. For the \mathcal{L}_{AS} loss, the trajectory error decreases for models A_6 to A_8 against \mathcal{L}_{MSE} , while the classification accuracy decreases. The same holds for the \mathcal{L}_H loss, where we can increase the classification accuracy up to 88.43%. Using *spatio-temporal* losses we are able to learn the shape

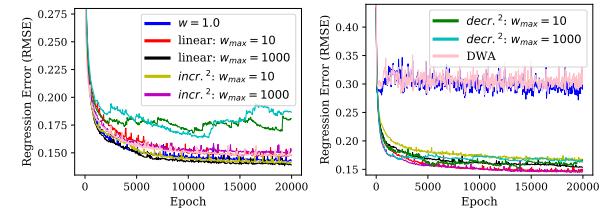


Figure 9: MTL strategy evaluation for \mathcal{L}_{MSE} combined with *spatio-temporal* \mathcal{L}_{PC} and *distribution-based* \mathcal{L}_{WAS_1} losses averaged over all architectures (baseline: blue).

of the characters (Fig. 7d and 7e), but at a wrong scale. A trajectory trained with the \mathcal{L}_{PC} is smoother (less variance) compared to \mathcal{L}_{CS} , yet with a lower accuracy in the classification. Our goal is to minimize the distance of the predicted trajectory while ensuring a smooth shape. Hence, we train the (distance) \mathcal{L}_{MSE} loss combined with the (spatio-temporal) \mathcal{L}_{PC} and \mathcal{L}_{CS} .

Combined Loss Functions. Fig. 8 compares all metrics based on the naive weighting. For the combination of \mathcal{L}_{MSE} and \mathcal{L}_{PC} , the regression loss improves over a model trained on \mathcal{L}_{MSE} only, while providing a smoother trajectory (Fig. 7f). The combined loss $\mathcal{L}_{MSE} + \mathcal{L}_{CS}$ does not scale the characters correctly (Fig. 7g) and gives a less smooth trajectory than the approach with either only \mathcal{L}_{MSE} (Fig. 7a), only \mathcal{L}_{PC} (Fig. 7d), or only \mathcal{L}_{CS} (Fig. 7e). We evaluate the \mathcal{L}_{WAS_p} loss when combined with the \mathcal{L}_{MSE} loss. For larger p , the predictions become more evenly distributed (as found by [23]). When choosing $p = 1$ and the naive weighting strategy (Fig. 9b) the trajectory prediction error is large (Fig. 7h), but can be significantly improved using alternative MTL strategies, as described in the following.

5.3. MTL Strategy Evaluation

We now compare all described MTL strategies (Sec. 3.3): naive weighting ($w=1$), linear as well as squared increase, squared weight decrease with maximal weighting of $w_{max}=\{10, 1000\}$, and DWA [40]. Fig. 9 shows MTL results for \mathcal{L}_{MSE} combined with \mathcal{L}_{PC} (Fig. 9a) and \mathcal{L}_{WAS_p} with $p=1$ (Fig. 9b). When combining \mathcal{L}_{WAS_1} with the \mathcal{L}_{PC} loss, the error further decreases for linear and squared weight increase. Squared weight decrease notably increases the error as the scaling of the shape diverges. With an optimal epoch-dependent weighting strategy the Pearson Correlation provides a suitable shape while still yielding to an improvement in accuracy. Combined with the \mathcal{L}_{WAS_1} loss, all weighting strategies significantly decrease the error (improvements between 0.16 to 0.18) in comparison to the naive approach (0.31), yet still preserve smooth predictions. When only training based on the \mathcal{L}_{MSE} loss, the distance is still smaller, but predicted trajectories are less smooth and not realistic. For the \mathcal{L}_{PC} loss combined with the \mathcal{L}_{MSE} loss DWA is worse than the linear or squared weight increase. As the loss of \mathcal{L}_{WAS_p} is higher than the loss of \mathcal{L}_{MSE} , DWA cannot optimize the training (Fig. 9b). We conclude, that the order of weight increase is important for combining metrics and that a slow weight increase is the best approach for jointly learning the classification and trajectory regression tasks.

5.4. Visual Dataset Evaluation

Loss Functions. The visual dataset is more challenging than the IMU dataset as the network needs to learn the transformation from the camera to the tablet coordinates and needs to identify the pen tip hover and touch data between strokes. This results in a much larger computing times, i.e., an average run time of 5.4s per epoch (see Appendix, Table 7). The baselines yield an error of 0.1360 for the trajectory regression task and 73.19% accuracy for the MTSC task (see Table 3). For the evaluation of MTL architectures, we can draw the same conclusions as in Sec. 5.1. Similar to the IMU dataset, for both, the \mathcal{L}_H and the \mathcal{L}_{AS} loss we (partly) observe a decrease in trajectory error at the cost of a worsening classification accuracy. The single *spatio-temporal* \mathcal{L}_{PC} loss increases the classification accuracy (80.51%), but yields improperly scaled characters. Through the combination of $\mathcal{L}_{MSE} + \mathcal{L}_{PC}$, we further decrease the trajectory error (0.1243) and increase the classification accuracy (81.64%) for architecture A_6 , while still providing a smooth trajectory. Neither the single *distribution-based* \mathcal{L}_{WAS_p} loss nor the combination with the \mathcal{L}_{MSE} loss can be used to improve the single and combined tasks. For more details, see Appendix, Sec. A.3.

Method	Inertial	Visual
MLSTM-FCN [31] w/ SE	89.33%	80.49%
w/o SE	88.41%	78.73%
TapNet [60]	89.02%	79.27%
Ours	89.51%	81.64%
	(A_4 , CS + CE)	(A_6 , MSE + PC + CE)

Table 4: Comparison of state-of-the-art MTSC methods with our approach on the inertial and visual datasets.

5.5. Comparison to MTSC Methods

We compare our method with MLSTM-FCN [31] and TapNet [60], two approaches achieving the highest classification accuracy on the well-known UEA MTS dataset [5]. For both techniques we interpolate the IMU data to 114 and the visual data to 71 timesteps. We evaluate various configurations of both networks in the line with suggestions by the authors. We optimized TapNet and searched for different configurations (with default parameters for the CNN, LSTM and random projection) and optimized the hyperparameters for training (such as different learning rates at 10^{-3} , 10^{-4} and 10^{-5}). For the MLSTM-FCN we achieved the best results with its squeeze-and-excitation block (see Table 4). TapNet achieves a better MTSC accuracy in comparison to MLSTM-FCN for the visual dataset. However, this still lacks behind our best reported results, which are both higher than TapNet and MLSTM-FCN. For the visual dataset our approach shows a larger margin and is notably better for certain loss combinations and architectures.

6. Conclusion

We addressed the problem of aligning a ground truth trajectory that is smooth over time by combining *distance-based* with *spatio-temporal* and *distribution-based* metrics. For the application of OnHW recognition of characters based on IMU and visual data sources, we significantly improved the trajectory prediction task, while still providing similar shapes. We proposed a framework of architectures and evaluated different MTL techniques. Through the combination of the trajectory prediction task with the subtly correlated MTS classification task we further improved the classification accuracy.

Acknowledgements

This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A (David Rügamer) and by the research program Human-Computer-Interaction through the project "Schreibtraine", Grant No. 16SV8228, as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGI-TAL II".

References

[1] Md. Shahinur Alam, Ki-Chul Kwon, Md. Ashraful Alam, Mohammed Y. Abbass, Shariar Md Imtiaz, and Nam Kim. Trajectory-Based Air-Writing Recognition Using Deep Neural Network and Depth Sensor. In *Sensors*, volume 20(376), Jan. 2020.

[2] Fevzi Alimoglu and Ethem Alpaydin. Combining Multiple Representations and Classifiers for Pen-based Handwritten Digit Recognition. In *Intl. Conf. on Document Analysis and Recognition (ICDAR)*, volume 2, Ulm, Germany, Aug. 1997.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. In *Intl. Conf. on Machine Learning (ICML)*, volume 70, pages 214–223, 2017.

[4] J. Arunnehru, A. K. Nandhana Davi, R. Raghul Sharan, and Poornima G. Nambiar. Human Pose Estimation and Activity Classification Using Machine Learning Approach. In *Intl. Conf. on Soft Computing and Signal Processing (ICSCSP)*, pages 113–123, Mar. 2020.

[5] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, and Paul Southam Eamonn Keogh. The UEA Multivariate Time Series Classification Archive, 2018. In *arXiv preprint arXiv:1811.00075*, Oct. 2018.

[6] Jonathan T. Barron. A General and Adaptive Robust Loss Function. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4326–4334, Long Beach, CA, June 2019.

[7] Vasileios Belagiannis, Christian Rupprecht, Gustavo Carneiro, and Nassir Navab. Robust Optimization for Deep Regression. In *Intl. Conf. on Computer Vision (ICCV)*, pages 2830–2838, Santiago de Chile, Chile, Dec. 2015.

[8] Michael J. Black and Anand Rangarajan. On the Unification of Line Processes, Outlier Rejection, and Robust Statistics with Applications in Early Vision. In *Intl. Journal of Computer Vision (IJCV)*, volume 19, page 57–91, July 1996.

[9] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2616–2625, 2018.

[10] Karl Bringmann. Why Walking the Dog Takes Time: Fréchet Distance has no Strongly Subquadratic Algorithms unless SETH fails. In *Symp. on Foundations of Computer Science (FOCS)*, Apr. 2014.

[11] Xingyu Cai, Tingyang Xu, Jin-Feng Yi, Junzhou Huang, and Sanguthevar Rajasekaran. DTWNet: a Dynamic Time Warping Network. In *Advances in Neural Information Processing Systems (NIPS)*, 2019.

[12] Junshen Kevin Chen, Wanze Xie, and Yutong (Kelly) He. Motion-based Handwriting Recognition. In *arXiv preprint arXiv:2101.06022*, Jan. 2021.

[13] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multi-task Networks. In *Intl. Conf. on Machine Learning (ICML)*, volume 80, pages 794–803, 2018.

[14] Samuel Cohen, Giulia Luise, Alexander Terenin, Brandon Amos, and Marc Peter Deisenroth. Aligning Time Series on Incomparable Spaces. In *Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 130, pages 1036–1044, 2021.

[15] Marco Cuturi and Mathieu Blondel. Soft-DTW: a Differentiable Loss Function for Time-Series. In *Intl. Conf. on Machine Learning (ICML)*, volume 70, pages 894–903, 2017.

[16] ShaiBen David and Reba Schuller. Exploiting Task Relatedness for Multiple Task Learning. In *Learning Theory and Kernel Methods*, volume 2777, pages 567–580, 2003.

[17] Thomas Deselaers, Daniel Keysers, and Henry A. Rowley. GyroPen: Gyroscopes for Pen-Input with Mobile Phones. In *Trans. on Human-Machine Systems*, Apr. 2015.

[18] Prafulla Dhariwal and Jeevana Inala. The Wasserstein Loss Function. Dec. 2015.

[19] Ian L. Dryden and Kanti V. Mardia. Statistical Shape Analysis with Applications in R. In *Wiley Series in Probability and Statistics*, 2016.

[20] Yunshu Du, Wojciech M. Czarnecki, Siddhant M. Jayakumar, Mehrdad Farajtabar, Razvan Pascanu, and Balaji Lakshminarayanan. Adapting Auxiliary Losses Using Gradient Similarity. In *arXiv preprint arXiv:1812.02224*, Nov. 2020.

[21] Yuchun Fang, Zhengyan Ma, Zhaoxiang Zhang, Xu-Yao Zhang, and Xiang Bai. Dynamic Multi-Task Learning with Convolutional Neural Network. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1668–1674, 2017.

[22] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep Learning for Time Series Classification: a Review. In *Data Mining and Knowledge Discovery*, July 2019.

[23] Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya-Polo, and Tomaso Poggio. Learning

with a Wasserstein Loss. In *Advances in Neural Information Processing Systems (NIPS)*, volume 2, page 2053–2061, Dec. 2015.

[24] Damien Garreau, Rémi Lajugie, Sylvain Arlot, and Francis Bach. Metric Learning for Temporal Sequence Alignment. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1817–1825, 2014.

[25] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. In *MIT Press*, 2016.

[26] Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor Forcing: A New Algorithm for Training Recurrent Networks. In *Advances in Neural Information Processing Systems (NIPS)*, Barcelona, Spain, 2016.

[27] Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic Task Prioritization for Multitask Learning. In *Europ. Conf. on Computer Vision (ECCV)*, pages 282–299, 2018.

[28] David Ha and Douglas Eck. A Neural Representation of Sketch Drawings. In *Intl. Conf. on Learning Representations (ICLR)*, Vancouver, BC, Canada, Apr. 2018.

[29] Yifan Hao and Huiping Cao. A New Attention Mechanism to Classify Multivariate Time Series. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1999–2005, 2020.

[30] Peter J. Huber. Robust Statistics. In *John Wiley and Sons*, New York, NY, 1981.

[31] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate LSTM-FCNs for Time Series Classification. In *Neural Network*, volume 116, pages 237–245, Aug. 2019.

[32] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 7482–7491, Salt Lake City, UT, 2018.

[33] Christopher Koellner, Marc Kurz, and Erik Sonnleitner. What Did You Mean? An Evaluation of Online Character Recognition Approaches. In *Intl. Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–6, 2019.

[34] Georg Kohl, Kiwon Um, and Nils Thuerey. Learning Similarity Metrics for Numerical Simulations. In *Intl. Conf. on Machine Learning (ICML)*, Vienna, Austria, Feb. 2020.

[35] Orhan Konak, Pit Wegner, and Bert Arnrich. IMU-Based Movement Trajectory Heatmaps for Human Activity Recognition. In *Sensors*, volume 20, Dec. 2020.

[36] Tuomo Korenius, Jorma Laurikkala, and Martti Juhola. On Principal Component Analysis, Cosine and Euclidean Measures in Information Retrieval. In *Information Science*, volume 177(22), pages 4893–4905, Nov. 2007.

[37] Vijay Kumar, Jitender Kumar Chhabra, and Dinesh Kumar. Performance Evaluation of Distance Metrics in the Clustering Algorithms. In *INFOCOMP*, volume 13(1), pages 38–51, June 2014.

[38] Changsheng Li, Junchi Yan, Fan Wei, Weishan Dong, Qingshan Liu, and Hongyuan Zha. Self-Paced Multi-Task Learning. In *Intl. Conf. on Association for the Advancement of Artificial Intelligence (AAAI)*, pages 2175–2181, 2017.

[39] Sicong Liang and Yu Zhang. A Simple General Approach to Balance Task Difficulty in Multi-Task Learning. In *arXiv preprint arXiv:2002.04792*, Feb. 2020.

[40] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-End Multi-Task Learning with Attention. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1871–1880, 2019.

[41] Shuangjun Liu, Naveen Sehgal, and Sarah Ostadbabas. Adapted Human Pose: Monocular 3D Human Pose Estimation with Zero Real 3D Pose Data. In *arXiv preprint arXiv:2105.10837*, May 2021.

[42] Keerthiram Murugesan and Jaime Carbonell. Self-Paced Multitask Learning with Shared Knowledge. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2522–2528, 2017.

[43] Nika Naghtabab, Cameron Musco, and Bo Waggoner. Toward a Characterization of Loss Functions for Distribution Learning. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, BC, Canada, 2019.

[44] Aviv Navon, Idan Achituv, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary Learning by Implicit Differentiation. In *Intl. Conf. on Learning Representations (ICLR)*, 2021.

[45] Dvir Ben Or, Michael Kolomenkin, and Gil Shabat. Generalized Quantile Loss for Deep Neural Networks. In *arXiv preprint arXiv:2012.14348*, Dec. 2020.

[46] Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 4(3), article 92, Cancún, Mexico, Sept. 2020.

4. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

- [47] Karl Pearson. Notes on the History of Correlation. In *Biometrika*, volume 13(1), pages 25–45, Oct. 1920.
- [48] Aaditya Ramdas, Nicolás García Trillo, and Marco Cuturi. On Wasserstein Two-Sample Testing and Related Families of Nonparametric Tests. In *Entropy*, volume 19(47), Jan. 2017.
- [49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Springer, LNCS, volume 9351, pages 234–241, 2015.
- [50] Maximilian Schrapel, Max-Ludwig Stadler, and Michael Rohs. Pentelligence: Combining Pen Tip Motion and Writing Sounds for Handwritten Digit Recognition. In *Conf. on Human Factors in Computing Systems*, number 131, pages 1–11, Apr. 2018.
- [51] Peter H. Schönemann. A Generalized Solution of the Orthogonal Procrustes Problem. In *Psychometrika*, volume 31(1), pages 1–10, Mar. 1966.
- [52] Ali Seyed Shirkhorshidi, Saeed Aghabozorgi, and Teh Ying Wah. A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data. In *PLOS ONE*, volume 10(12), Dec. 2015.
- [53] Anuj Srivastava and Eric P. Klassen. Functional and Shape Data Analysis. In *Springer Series in Statistics*, 2016.
- [54] Marc Strickert, Frank-Michael Schleif, and Udo Seifert. Derivatives of Pearson Correlation for Gradient-based Analysis of Biomedical Data. In *Inteligencia Artificial*, volume 12(37), pages 37–44, Mar. 2008.
- [55] Abdel Aziz Taha and Allan Hanbury. An Efficient Algorithm for Calculating the Exact Hausdorff Distance. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 37(11), Nov. 2015.
- [56] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. In *Journal of Machine Learning Research (JMLR)*, pages 2579–2605, Nov. 2008.
- [57] Cédric Villani. Optimal Transport, Old and New. In *Springer*, Dec. 2006.
- [58] Jeen-Shing Wang, Yu-Liang Hsu, and Cheng-Ling Chu. Online Handwriting Recognition Using and Accelerometer-Based Pen Device. In *Intl. Conf. on Computer Science and Engineering (CSE)*, 2013.
- [59] Hanyuan Zhang, Xingyu Zhang, Qize Jiang, Baihua Zheng, Zhenbang Sun, Weiwei Sun, and Changhu Wang. Trajectory Similarity Learning with Auxiliary Supervision and Optimal Matching. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 3209–3215, 2020.
- [60] Xuchao Zhang, Yifeng Gao, Jessica Lin, and Chang-Tien Lu. TapNet: Multivariate Time Series Classification with Attentional Prototypical Network. In *Intl. Conf. on Association for the Advancement of Artificial Intelligence (AAAI)*, pages 6845–6852, 2020.
- [61] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial Landmark Detection by Deep Multi-Task Learning. In *Europ. Conf. on Computer Vision (ECCV)*, pages 94–108, 2014.
- [62] PYi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J. Leon Zhao. Exploiting Multi-Channels Deep Convolutional Neural Networks for Multivariate Time Series Classification. In *Frontiers of Computer Science*, volume 10, pages 96–112, Sept. 2015.

A. Appendix

We present criteria for loss functions and an overview of metrics in Section A.1. We propose more details about the architectures in Sec. A.2, and present more results in Sec. A.3. Sec. A.4 describes a dataset-specific feature embedding analysis.

A.1. Loss Functions

In the following, we state all loss functions that we used for our methodology. We describe the cross-entropy loss for the Multivariate Time Series Classification (MTSC) task. Next, we present criteria for the trajectory regression task. Finally, we propose *distance-based*, *spatio-temporal* and *distribution-based* loss functions.

MTSC Task: Cross-entropy Loss. For the MTSC task, the cross-entropy loss [25] is defined by

$$\mathcal{L}_{CE}(\mathcal{U}, \mathcal{V}) = -\frac{1}{n} \sum_{i=1}^n v_i \log \hat{v}_i, \quad (2)$$

where the Multivariate Time Series (MTS) $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $m \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l}), i \in \{1, \dots, m\}$. m is the length of the time series and l is the number of dimensions. Each MTS is associated with a class label $v \in \Omega$ from a pre-defined label set Ω . The training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_n\} \in \mathbb{R}^{n \times m \times l}$, where n is the number of time series, and the corresponding labels $\mathcal{V} = \{v_1, \dots, v_n\} \in \Omega^n$ [60]. The MTSC task is to predict an unknown class label \hat{v} for a given MTS.

Trajectory Regression Task: Criteria. For the trajectory regression task it is given a ground truth time series $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_m\} \in \mathbb{R}^{m \times d}$. The goal is to predict a time series $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \in \mathbb{R}^{n \times d}$, such that \mathcal{X} is closely aligned to \mathcal{Y} . In the following, we consider $\mathbf{r}_i = \mathbf{y}_i - \mathbf{x}_i$ be the residual between \mathcal{X}_i and \mathcal{Y}_i . We consider a (differentiable) substitution-cost function $\mathcal{L} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$ to learn the trajectory regression task. All metrics to be used in a neural network have to obey the following criteria, where $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \in \mathbb{R}$ [34]:

$$\mathcal{L}(\mathcal{X}, \mathcal{Y}) \geq 0 \quad (\text{non-negativity}) \quad (I)$$

$$\mathcal{L}(\mathcal{X}, \mathcal{Y}) = \mathcal{L}(\mathcal{Y}, \mathcal{X}) \quad (\text{symmetry}) \quad (II)$$

$$\mathcal{L}(\mathcal{X}, \mathcal{Y}) \leq \mathcal{L}(\mathcal{X}, \mathcal{Z}) + \mathcal{L}(\mathcal{Z}, \mathcal{Y}) \quad (\text{triangle inequ.}) \quad (III)$$

$$\mathcal{L}(\mathcal{X}, \mathcal{Y}) = 0 \Leftrightarrow \mathcal{X} = \mathcal{Y} \quad (\text{ident. of indiscernibles}) \quad (IVa)$$

It is difficult to make accurate predictions about the injectivity as floating points operations and approximation errors lead to a distance of zero for slightly different inputs.

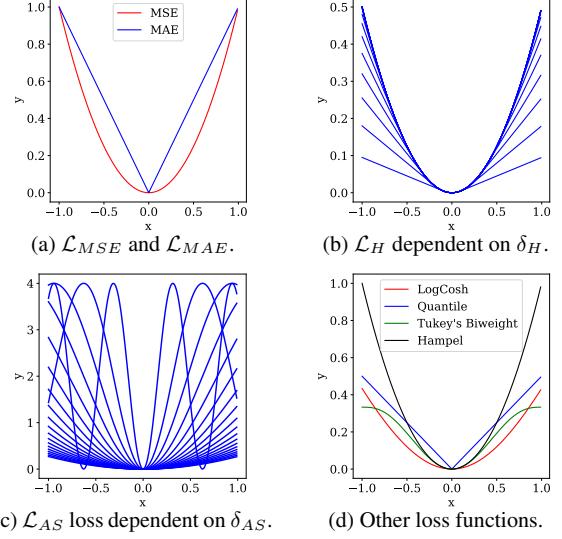


Figure 10: Sample weighting based on different *distance-based* metrics.

Hence, Equ. (IVa) can be formulated as a *pseudometric* with a relaxed identity of indiscernibles where $\tilde{\mathcal{X}} \in \mathbb{R}$:

$$\mathcal{L}(\tilde{\mathcal{X}}, \tilde{\mathcal{X}}) = 0. \quad (IVb)$$

Trajectory Regression Task: Distance-based Loss Functions. We consider the Mean Squared Error

$$\mathcal{L}_{MSE}(\mathcal{X}, \mathcal{Y}) = \frac{1}{n} \|\mathcal{Y} - \mathcal{X}\|_2^2 = \frac{1}{n} \sum_{i=1}^n \mathbf{r}_i^2 \quad (3)$$

with L_2 -norm $\|\cdot\|_2$ (see Fig. 10a). The derivative of the \mathcal{L}_{MSE} loss is $\frac{\partial}{\partial \mathcal{X}} \mathcal{L}_{MSE}(\mathcal{X}, \mathcal{Y}) = -\frac{2}{n} \sum_{i=1}^n \mathbf{r}_i$. The Mean Absolute Error (MAE) is

$$\mathcal{L}_{MAE}(\mathcal{X}, \mathcal{Y}) = \frac{1}{n} \|\mathcal{Y} - \mathcal{X}\|_1 = \frac{1}{n} \sum_{i=1}^n |\mathbf{r}_i| \quad (4)$$

with the L_1 -norm $\|\cdot\|_1$. Its derivative is $\frac{\partial}{\partial \mathcal{X}} \mathcal{L}_{MAE}(\mathcal{X}, \mathcal{Y}) = 1/n \sum_{i=1}^n \text{sign}(\mathbf{r}_i)$. The Huber loss [30]

$$\mathcal{L}_H(\mathcal{X}, \mathcal{Y}, \delta_H) = \sum_{i:|\mathbf{r}_i| \leq \delta_H} \frac{1}{2} (\mathbf{r}_i)^2 + \sum_{i:|\mathbf{r}_i| > \delta_H} \delta_H |\mathbf{r}_i| - \frac{1}{2} \delta_H^2 \quad (5)$$

is less sensitive to outliers, but depends on the hyperparameter δ_H (see Fig. 10b). The derivative of the Huber loss is

$$\frac{\partial}{\partial \mathcal{X}} \mathcal{L}_H(\mathcal{X}, \mathcal{Y}, \delta_H) = - \sum_{i:|\mathbf{r}_i| \leq \delta_H} \mathbf{r}_i - \sum_{i:|\mathbf{r}_i| > \delta_H} \delta_H \text{sign}(\mathbf{r}_i). \quad (6)$$

4. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

Similar, the Andrew's Sine loss [8] is

$$\mathcal{L}_{AS}(\mathcal{X}, \mathcal{Y}, \delta_{AS}) = \sum_{i:|\mathbf{r}_i| \leq 1}^n 4 \sin \left(\frac{\mathbf{r}_i}{2\delta_{AS}} \right)^2 + \sum_{i:|\mathbf{r}_i| > 1}^n 1, \quad (7)$$

with hyperparameter δ_{AS} (see Fig. 10c) with the derivative

$$\frac{\partial}{\partial \mathcal{X}} \mathcal{L}_{AS}(\mathcal{X}, \mathcal{Y}, \delta_{AS}) = \sum_{i:|\mathbf{r}_i| \leq 1}^n \frac{2}{\delta_{AS}} \sin \left(\frac{\mathbf{r}_i}{\delta_{AS}} \right). \quad (8)$$

Trajectory Regression Task: Spatio-temporal Loss Functions.

We define the Cosine Similarity by

$$\mathcal{L}_{CS}(\mathcal{X}, \mathcal{Y}) = 1 - \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2}. \quad (9)$$

The Cosine Similarity is a proper metric as it satisfies the requirements $\mathcal{L}_{CS}(\mathcal{X}, \mathcal{Y}) \geq 0$ (I), $\mathcal{L}_{CS}(\mathcal{X}, \mathcal{Y}) = \mathcal{L}_{CS}(\mathcal{Y}, \mathcal{X})$ (II), and $\mathcal{L}_{CS}(\mathcal{X}, \mathcal{X}) = 0$ (IVb). Under certain conditions the triangle equality (III) is not fulfilled [36]. This loss function is a measure of similarity between two non-zero vectors of an inner product space, but is not invariant to shifts. The Pearson Correlation loss [47] $\mathcal{L}_{PC}(\mathcal{X}, \mathcal{Y}) = \mathcal{L}_{CS}(\mathcal{X} - \bar{\mathcal{X}}, \mathcal{Y} - \bar{\mathcal{Y}})$, in contrast, is invariant to shifts. This means, when \mathcal{X} is transformed by $a + b\mathcal{X}$ and \mathcal{Y} is transformed by $c + d\mathcal{Y}$, where a, b, c and d are constants ($b, d > 0$), the Pearson Correlation coefficient is invariant in location and scale in the two variables. The Pearson Correlation loss is defined by

$$\mathcal{L}_{PC}(\mathcal{X}, \mathcal{Y}) = 1 - \frac{s_{xy}}{s_x \cdot s_y} = 1 - \frac{(\mathbf{x} - \bar{\mathbf{x}}) \cdot (\mathbf{y} - \bar{\mathbf{y}})}{\|\mathbf{x} - \bar{\mathbf{x}}\|_2 \|\mathbf{y} - \bar{\mathbf{y}}\|_2}. \quad (10)$$

with the sample mean $\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$. Analogously for $\bar{\mathbf{y}}$. The covariance is $s_{xy} = \frac{1}{n} (\mathbf{x} - \bar{\mathbf{x}}) \cdot (\mathbf{y} - \bar{\mathbf{y}})$, and the variance of the features is $s_x^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})^2$, analogously for s_y^2 . The partial derivative of the Pearson Correlation [54] regarding \mathbf{x} is

$$\frac{\partial}{\partial \mathbf{x}} \mathcal{L}_{PC}(\mathcal{X}, \mathcal{Y}) = \frac{(\mathbf{y} - \bar{\mathbf{y}}) - \frac{s_{xy}}{s_y} \cdot (\mathbf{x} - \bar{\mathbf{x}})}{s_x \cdot s_y}. \quad (11)$$

Further alternative *distance*-based metrics are, e.g., the LogCosh, the Quantile [45], the Tukey's Biweight [7], the Hampel [8] and the Geman McClure metric [6] (see Fig. 10d). For more information, see [37] for *distance*-based metrics and [52] for *spatio-temporal* metrics.

Trajectory Regression Task: Distribution-based Loss Functions. We use the *distribution*-based loss function, i.e., the Wasserstein distance [23], defining a distance between two probability distributions on a given metric space \mathcal{M} and representing the cost δ of an optimal mass transportation problem. Optimal transport can be used to compare probability measures in metric spaces. There exists

some \mathcal{X}_0 in \mathcal{M} such that the Wasserstein space of order p is defined as

$$P_p(\mathcal{M}) := \left\{ \mu \in P(\mathcal{M}); \int_{\mathcal{M}} \delta(\mathcal{X}, \mathcal{X}_0)^p d\mathcal{X} < \infty \right\}. \quad (12)$$

The p^{th} Wasserstein distance between two probability measures μ and ν is defined as

$$\begin{aligned} W_p(\mu, \nu) &:= \left(\inf_{\gamma \in \Gamma(\mu, \nu)} \int_{\mathcal{M} \times \mathcal{M}} \delta(\mathcal{X}, \mathcal{Y})^p d\gamma(\mathcal{X}, \mathcal{Y}) \right)^{\frac{1}{p}} \\ &= \inf \left\{ [\mathbb{E}[d(X, Y)^p]]^{\frac{1}{p}}, \text{ law}(X) = \mu, \text{ law}(Y) = \nu \right\}, \end{aligned} \quad (13)$$

with the collection of all probability measures $\Gamma(\mu, \nu)$ on $\mathcal{M} \times \mathcal{M}$ and $p \in [1, \infty)$. $\mathbb{E}[X]$ denotes the expected value of a random variable X . We consider the classical case where the metric is the Euclidean metric in space $\mathbb{R}^d \subset \mathcal{M}$, and hence, $\delta(\mathcal{X}, \mathcal{Y}) = \|\mathcal{X} - \mathcal{Y}\|$. For all subsets $P \subset \mathbb{R}^d$, we have $\gamma(P \times \mathbb{R}^d) = \mu(P)$ and $\gamma(P \times \mathbb{R}^d) = \nu(P)$. [48] The W_1 distance is also called the Kantorovich-Rubinstein distance. The Wasserstein distance satisfies the criteria (I) to (IVa): It holds the non-negativity criteria $W_p(\mu, \nu) \geq 0$ (I), and the symmetry criteria $W_p(\mu, \nu) = W_p(\nu, \mu)$ (II). Assume that $W_p(\nu, \mu) = 0$, then there exists a transference plan that is concentrated on the diagonal, and it holds $\mathcal{X} = \mathcal{Y}$ (IVa). Furthermore, let μ_1, μ_2 and μ_3 be probability measures on $\mathcal{M} \times \mathcal{M}$, and (T_1, T_2) , respectively (Q_2, Q_3) , be an optimal coupling of (μ_1, μ_2) , respectively of (μ_2, μ_3) . There exist random variables (T'_1, T'_2, T'_3) with $\text{law}(T'_1, T'_2) = \text{law}(T_1, T_2)$ and $\text{law}(T'_2, T'_3) = \text{law}(Q_2, Q_3)$, such that

$$\begin{aligned} W_p(\mu_1, \mu_3) &\leq (\mathbb{E}[d(T'_1, T'_3)^p])^{\frac{1}{p}} \leq \\ &\leq (\mathbb{E}[d(T'_1, T'_2) + d(T'_2, T'_3)]^p)^{\frac{1}{p}} \leq \\ &\leq (\mathbb{E}[d(T'_1, T'_2)^p])^{\frac{1}{p}} + (\mathbb{E}[d(T'_2, T'_3)^p])^{\frac{1}{p}} = \\ &= W_p(\mu_1, \mu_2) + W_p(\mu_2, \mu_3), \end{aligned} \quad (14)$$

and the triangle inequality holds (III). The duality formula for the Kantorovich-Rubinstein distance is

$$W_1(\mu, \nu) = \sup_{\|\psi\|_{Lip} \leq 1} \left\{ \int_{\mathcal{M}} \psi d\mu \int_{\mathcal{M}} \psi d\nu \right\}, \quad (15)$$

for any μ, ν in the Wasserstein space $P_1(\mathcal{M})$. The Wasserstein distance W_1 of order 1 is the weakest of all, and hence, is easier to bound. The Wasserstein distance has the ability to capture weak convergence precisely and are rather strong as they take care of large distances in $\mathcal{M} \times \mathcal{M}$. [57] For more information, see [43].

Summary. For the classification task, we use the \mathcal{L}_{CE} loss function (2). For the regression task, we use a combination of the *distance*-based loss functions \mathcal{L}_{MSE} (3), \mathcal{L}_{MAE}

Network	Total	Trunk	Regr.	Class.
Only regression (A_0)	117,052	96,852	20,200	-
Only classification (A_1)	133,735	117,051	-	16,683
Class. for regr. (A_2)	190,535	117,052	56,800	16,683
Latest split (A_3)	133,735	96,852	20,200	16,683
Late split (A_4)	153,935	96,852	20,200	36,883
Split after LSTM (A_5)	194,935	86,352	30,700	77,883
Split after 2. Drop. (A_6)	260,935	20,352	96,700	143,883
Split after 1. Drop. (A_7)	277,639	3,648	113,404	160,587
Separate heads (A_8)	281,287	0	117,052	164,235

Table 5: Parameters of the inertial-based models.

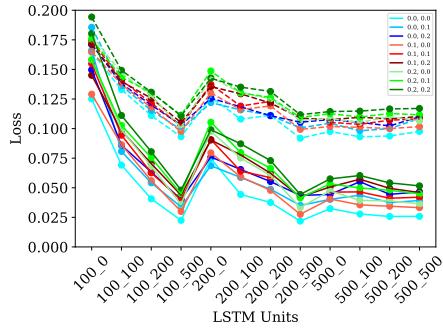


Figure 11: Grid search for an optimal number of LSTM units and dropout for the visual-based CNN. Continuous line: training loss. Dashed line: validation error.

(4), \mathcal{L}_H (5) and \mathcal{L}_{AS} (7), *spatio-temporal* loss functions \mathcal{L}_{CS} (9) and \mathcal{L}_{PC} (10), and the *distribution-based* Wasserstein function \mathcal{L}_{WAS_p} (13).

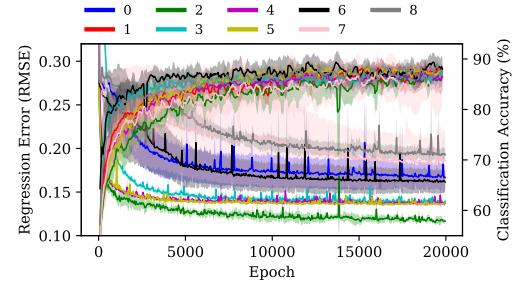
A.2. MTL Network Architectures

The general overview of the framework is given in Fig. 2 (Sec. 3). The input is for the IMU-based and the visual-based *OnHW* dataset a MTS that differs regarding its input size. For the IMU-based dataset, the input are the 13 channels of the accelerometers, gyroscope, magnetometer and force sensor. The number of timesteps depends on the sample length. For the visual dataset, the input is the two dimensional trajectory of the pen tip in camera coordinates. What follows, is a CNN trunk, a classification head, and a regression head. The classification head is used for the MTSC task by predicting a class label with the cross-entropy loss. The regression head is used for the trajectory regression task that predicts a MTS that represents the trajectory of the written character. The loss function for this task is a combination of *distance-based*, *spatio-temporal*, and *distribtuion-based* metrics.

The number of trainable parameters in the neural network trunk and in task-specific heads is important. We address the problem in the following. We construct for each dataset nine architectures with different split points. Architectures A_0 and A_1 are Single Task Learning (STL) CNNs for the MTSC task and the trajectory prediction task. Archi-

Network	Total	Trunk	Regr.	Class.
Only regression (A_0)	1,342,638	1,322,438	20,200	-
Only classification (A_1)	1,359,321	1,342,638	-	16,683
Class. for regr. (A_2)	1,416,121	1,342,638	56,800	16,683
Latest split (A_3)	1,359,321	1,355,804	20,200	16,683
Late split (A_4)	1,379,521	1,322,438	20,200	36,883
Split after 1. LSTM (A_5)	1,619,921	1,082,038	260,600	277,283
LSTM in traj. head (A_6)	1,459,521	1,082,038	260,600	116,883
LSTM in class. head (A_7)	1,459,521	982,038	100,200	377,283
Split after 1. Drop. (A_8)	2,701,959	76	1,342,600	1,359,283

Table 6: Parameters of the visual-based models.

Figure 12: Evaluation for CNN architectures (A_0 - A_8) trained with the \mathcal{L}_{MSE} and \mathcal{L}_{CE} loss averaged over all trainings.

tures A_2 to A_8 combine both tasks by MTL. All IMU-based architectures are given in Fig. 3. All visual-based architectures are given in Fig. 4, where we search for the optimal number of LSTM units and dropouts (see Fig. 11). We choose a combination of 500 and 100 LSTM units, and two dropout layers of 20%. An overview of all architectures and its number of trainable parameters is given in Table 5 and in Table 6. The number of total parameters increases for an early split point compared to late split points for both networks. A regression head with one *dense* layer of 200 neurons has 20,200 parameters, while a classification head with one *dense* layer of 83 neurons has 16,683 parameters. Fig. 12 compares the training loss of all architectures for the regression and classification tasks.

A.3. Evaluation Results

Hyperparameter Search for the Inertial-based Architectures. For the alternative *distance*-based metrics, i.e., Andrew's Sine and Huber, we search for the hyperparameters δ_H and δ_{AS} in $\{0.1, 0.2, \dots, 2.0, 2.5, \dots, 5.0\}$ using a grid search (Fig. 14a). For \mathcal{L}_{AS} we choose $\delta_{AS} = 0.3$, and for \mathcal{L}_H we choose $\delta_H = 4.0$ for follow-up training. A large δ_H tends to weight outliers more, while a small outlier rejection has a higher standard deviation. We also search for the hyperparameter $p \in \{1, 2, 3, 4\}$ of the *distribution-based* loss \mathcal{L}_{WAS_p} (Fig. 14a) and choose $p = 1$ for follow-up training.

4. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach

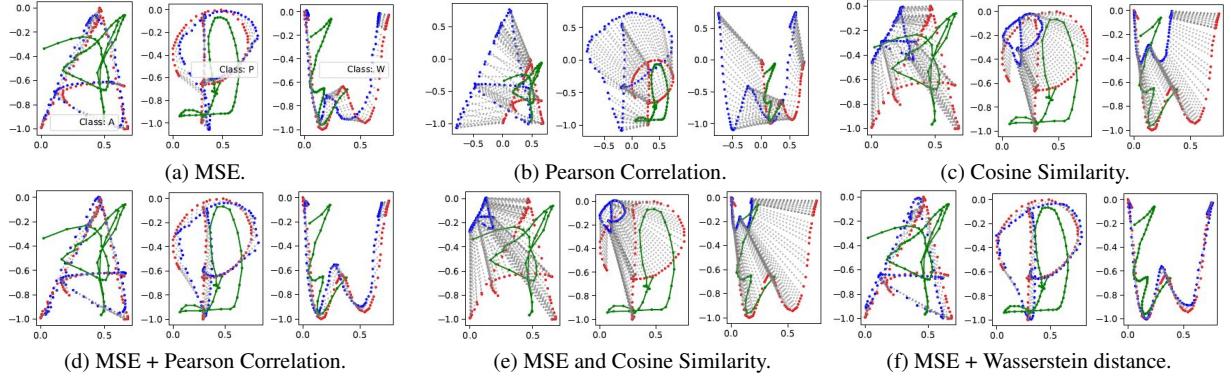


Figure 13: Trajectory prediction (blue) against the ground truth trajectory (red) of the characters 'A', 'P' and 'W' based on visual data (green) as MTS input preprocessed with U-Net [49].

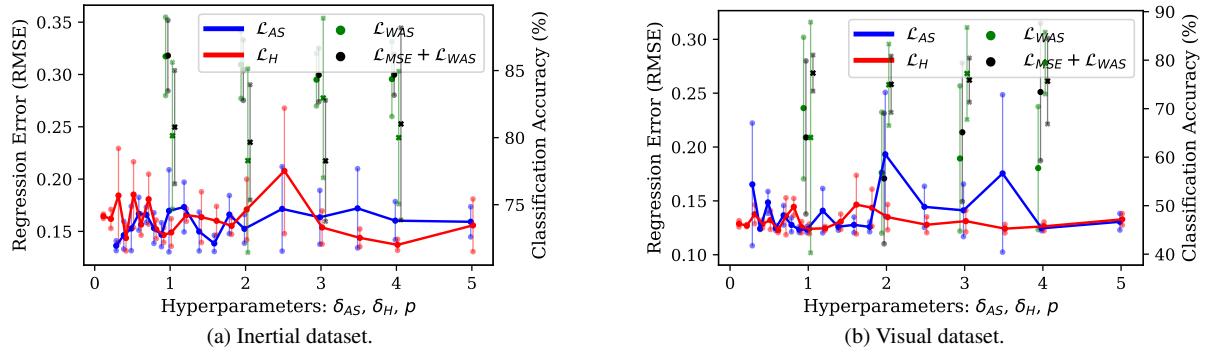


Figure 14: Grid search for δ_{AS} , δ_H and p averaged over all architectures. For \mathcal{L}_{WAS_p} : left: trajectory error, right: classification accuracy.

Hyperparameter Search for Visual-based Architectures.

As for the inertial-based architectures, we search for the optimal hyperparameter of alternative distance-based metrics using a grid search (see Fig. 14b). For \mathcal{L}_{AS} we choose $\delta_{AS} = 2.5$, and for \mathcal{L}_H we choose $\delta_H = 2.0$ for follow-up training. We also search for the hyperparameter p in the *distribution*-based loss \mathcal{L}_{WAS_p} and choose $p = 4$ for follow-up training as it achieves the highest classification accuracy for both the single \mathcal{L}_{WAS_4} loss and the combination of \mathcal{L}_{MSE} and \mathcal{L}_{WAS_4} .

Camera-based reconstruction. In Fig. 13 we propose additional trajectory reconstruction results based on the visual dataset. We come to similar conclusions as for the inertial-based dataset. While the CNN trained with the \mathcal{L}_{MSE} loss combined with the \mathcal{L}_{PC} loss (Fig. 13d) predicts a smoother trajectory than the single *distance*-based loss functions (Fig. 13a), the $\mathcal{L}_{MSE} + \mathcal{L}_{WAS}$ loss allows a proper training (Fig. 13f).

Loss function	IMU-based CNN	Visual-based CNN
\mathcal{L}_{MSE}	0.3052 ± 0.0401	5.8961 ± 0.0533
\mathcal{L}_H	0.2971 ± 0.0401	5.8154 ± 0.1067
\mathcal{L}_{AS}	0.2993 ± 0.0399	5.8349 ± 0.1259
\mathcal{L}_{PC}	0.3027 ± 0.0394	4.4017 ± 0.0924
\mathcal{L}_{CS}	0.3001 ± 0.0395	4.4246 ± 0.0488
\mathcal{L}_{WAS_1}	0.2994 ± 0.0395	4.4538 ± 0.0996
$\mathcal{L}_{MSE} + \mathcal{L}_{PC}$	0.3078 ± 0.0397	5.9077 ± 0.0610
$\mathcal{L}_{MSE} + \mathcal{L}_{CS}$	0.2984 ± 0.0387	5.8790 ± 0.1133
$\mathcal{L}_{MSE} + \mathcal{L}_{WAS_1}$	0.3096 ± 0.0396	5.9607 ± 0.1253

Table 7: Overview of inference times in seconds (s) (mean and standard deviation over all epochs) for architecture A_0 .

Training Times. For all loss combinations, we present inference times (Table 7). While the visual-based CNN takes 5.3971s for each epoch (on average), the IMU-based CNN only takes 0.3022s. The differences between the loss functions are small for the IMU-based architectures. From the visual-based CNNs we can see that the *spatio-temporal* and *distribution*-based loss functions (4.4267s) are less computationally intense compared to the *distance*-based loss functions (5.8488s). The marginally increased computing time for all loss combinations (5.9158s) is negligible.

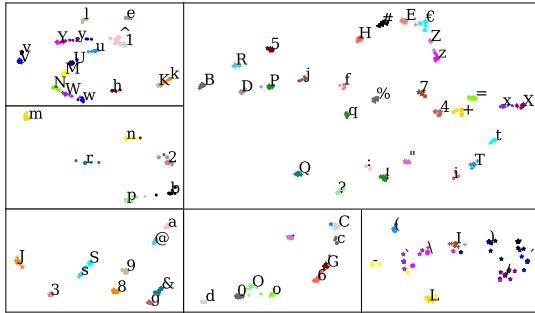


Figure 15: Visualization of the network embedding of the last layer based on the t-SNE algorithm by [56].

A.4. Dataset Feature Embedding Evaluation

Fig. 15 illustrates the challenges of the inertial dataset using a 300 dimensional feature embedding. The figure represents the feature embedding based on the t-SNE algorithm [56] (initial dimension of 301, perplexity of 30, an initial momentum of 0.5, and a final momentum of 0.8) and incorporates all 83 classes, i.e., capital and small characters, numbers and symbols. As it can be seen, the classes can be well separated. The main difficulty is to differentiate capital and small letters, e.g., 'V' and 'v', 'K' and 'k', and 'X' and 'x', as these differ only in the size and not in the number of strokes. Naturally, the embeddings of the letters 'O', 'o' and 'ø' are very close to each other. Furthermore, the classes 'G', '6', 'C', 'c' and 'C' are challenging to distinguish. These are one of the most frequent errors of the networks for the MTSC task.

Chapter 5

Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift

Contributing Article

Andreas Klaß, Sven M. Lorenz, Martin W. Lauer-Schmaltz, David Rügamer, Bernd Bischl, Christopher Mutschler, and Felix Ott. Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift. In *IJCAI-ECAI International Workshop on Spatio-Temporal Reasoning and Learning (STRL)*, volume 3190, Vienna, Austria, July 2022.

Publisher Website

<http://ceur-ws.org/Vol-3190/>

Copyright License

This work is licensed under a Creative Commons Attribution International 4.0 License (<https://creativecommons.org/licenses/by/4.0/>). © Copyright held by the owner/author(s).

Author Contributions¹⁵

This paper is based on the consulting project carried out by Andreas Klaß and Sven M. Lorenz, which was supervised by David Rügamer and Felix Ott. Andreas Klaß, Sven M. Lorenz and Felix Ott are the corresponding authors of this paper. Andreas Klaß and Sven

M. Lorenz contributed equally with the conceptualization, methodology, software, validation, formal analysis, hyperparameter tuning, investigation, writing of the original draft and editing, and visualization. Martin W. Lauer-Schmaltz performed an initial data analysis, software, implementation of metric functions and validation parameters (i.e., coverage), and implementation of the Bayes by Backpropagation method (not included in this paper), and contributed with writing (i.e., review and editing). David Rügamer contributed with the conceptualization of the paper (i.e., ideas and aims), formal analysis, writing (i.e., thorough review and editing), and supervision. Bernd Bischl contributed by supervision. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), supervision, and funding acquisition of the project “Schreibtrainer” (grant number 16SV8228) by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”. Felix Ott contributed by the conceptualization of the paper (i.e., ideas and aims), model training (execution), validation, formal analysis, investigation, data curation (i.e., maintaining research data and maintaining the website), writing of the original draft, reviewing and editing, visualization, and project administration. The evaluation results are based on the consulting project by Andreas Klaß and Sven M. Lorenz, but the paper has been rewritten (with an overlap of Section 3 “Methodological Background”), figures have been recreated, and main contributions and interpretations have been exposed. This paper was presented by Andreas Klaß at the IJCAI-ECAI International Workshop on Spatio-Temporal Reasoning and Learning (STRL) in July 2022 in Vienna, Austria.

Datasets and Source Code

This paper uses the OnHW-chars dataset from right-handed writers and left-handed writers. The source code to reproduce evaluation results is publicly available at:
<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>
https://gitlab.cc-asp.fraunhofer.de/ottf/uq_time_series

Statement about Recent, Related Research¹⁷

In the contributing paper (Klaß et al., 2022), the authors evaluate a widely used uncertainty decomposition by decomposing the total uncertainty into aleatoric and epistemic uncertainty. Recently, new measures based on credal sets have been proposed in the literature (Hüllermeier et al., 2022), which may be suitable for evaluating OnHW datasets in future research and compared with the commonly used decomposition. While the paper (Klaß et al., 2022) employs Deep Ensembles and Stochastic Weight Averaging-Gaussian (SWAG) to estimate the prediction variance, Raichur et al. (2022) use Monte Carlo dropout for global navigation satellite systems (GNSS) interference classification. It is important to note that the performance of the uncertainty quantification method is dependent on the dataset.

Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift

Andreas Klaß^{1,2,†}, Sven M. Lorenz^{1,2,†}, Martin W. Lauer-Schmalz⁴, David Rügamer^{2,3}, Bernd Bischl², Christopher Mutschler¹ and Felix Ott^{1,2}

¹*Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits IIS*

²*LMU Munich, Munich, Germany*

³*RWTH Aachen, Aachen, Germany*

⁴*Technical University of Denmark*

Abstract

For many applications, analyzing the uncertainty of a machine learning model is indispensable. While research of uncertainty quantification (UQ) techniques is very advanced for computer vision applications, UQ methods for spatio-temporal data are less studied. In this paper, we focus on models for online handwriting recognition, one particular type of spatio-temporal data. The data is observed from a sensor-enhanced pen with the goal to classify written characters. We conduct a broad evaluation of aleatoric (data) and epistemic (model) UQ based on two prominent techniques for Bayesian inference, Stochastic Weight Averaging-Gaussian (SWAG) and Deep Ensembles. Next to a better understanding of the model, UQ techniques can detect out-of-distribution data and domain shifts when combining right-handed and left-handed writers (an underrepresented group).

1. Introduction

Traditional machine learning (ML) algorithms assume training and test datasets to be *independently and identically distributed* [1, 2]. For many real-world applications, data often changes over time and space, and hence, training and test data originate from different distributions. This can cause ML models to fail due to a *domain shift* between training and test data [1]. Transfer learning [3, 4] and domain adaptation [5, 6] techniques can compensate for this domain shift. A first step in adapting for this domain shift is its detection, e.g., by having reliable uncertainty estimates of the model predictions [7]. Thus, to estimate the uncertainty of the model, a robust uncertainty quantification (UQ) technique is required that runs in real-time.

Approximate Bayesian Inference Techniques. In the field of deep learning (DL), UQ has lately seen a steep increase in interest. Recently, many promising methods have been proposed such as Variational Online Gauss-Newton (VOGN) [8], Stochastic Weight Averaging-Gaussian (SWAG) [9], Bayes by Backpropagation (BBB)

[10], and Laplace Approximation [11]. Another widely used technique are Deep Ensembles [12], which often yield well-calibrated models while being relatively easy to implement.

Decomposing Uncertainty. Several ways for estimating and decomposing uncertainty have been proposed. A common distinction is made between *aleatoric* uncertainty, which refers to the variability in the data, and *epistemic* uncertainty, which is the model's uncertainty caused by a lack of knowledge [13]. Building on [14], [15] argue that neural networks (NNs) for classification are basically generalized linear models with error structure of multinomial and composite link functions. Hence, to acknowledge that the variance of a multinomial outcome is a function of the mean outcome, they propose to directly compute the variability in the softmax outputs. Another method to dissect total predictive uncertainty has been put forward by [16] and similarly by [17] who propose to extract epistemic and aleatoric uncertainties from the predictive distribution of a Bayesian NN by calculating the entropy and mutual information. For an extensive survey of related approaches, see [18].

UQ for OnHW. UQ techniques have been broadly evaluated in computer vision applications such as image classification [14], i.e., optical character recognition (OCR), but methods have rarely been evaluated on spatio-temporal datasets [19]. OCR is concerned with offline handwriting recognition from images. In contrast, online handwriting (OnHW) recognition works on different types of spatio-temporal signals and can make use of

STRL '22: First International Workshop on Spatio-Temporal Reasoning and Learning, July 24, 2022, Vienna, Austria

[†] These authors contributed equally.

✉ a.klass@campus.lmu.de (A. Klaß); sven.lorenz@campus.lmu.de (S. M. Lorenz); mwola@dtu.dk (M. W. Lauer-Schmalz); david.ruegamer@stat.uni-muenchen.de (D. Rügamer); bernd.bischl@stat.uni-muenchen.de (B. Bischl); christopher.mutschler@iis.fraunhofer.de (C. Mutschler); felix.ott@iis.fraunhofer.de (F. Ott)

 © 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

temporal information such as writing speed and direction [20]. While many recording systems make use of a stylus pen together with a touch screen surface, sensor-enhanced pens, e.g., [21, 22, 23, 24], based on inertial measurement units (IMUs) enable new applications. These pens stream data from accelerometer, gyroscope, magnetometer and force sensors in real-time represented as spatio-temporal multivariate time-series (MTS). The advantage of exploiting this temporal information is the ability to better distinguish between similarly shaped letters from dynamic information (number of strokes etc.). Spatio-temporal data can further help to identify certain characteristics in the data. [25], e.g., showed the domain shift between right-handed and left-handed writers by analyzing feature embeddings of their model for OnHW data.

Contribution. In this paper we evaluate the uncertainty of OnHW model predictions with SWAG [9] and Deep Ensembles [12] for spatio-temporal reasoning, assessment of out-of-distribution detection, and pattern and failure recognition. We use uncertainty decompositions based on the method by [15] and [16] to evaluate the UQ techniques. Our claims are further supported by utilizing confidence and accuracy metrics to estimate the expected calibration error (ECE) [26]. For an OnHW task with domain shift between right- and left-handed writers, we evaluate uppercase, lowercase and combined character classification tasks. Our source code will be available upon publication.¹

The remainder of the paper is organized as follows. Section 2 discusses related work. In Section 3, we describe the background of Bayesian modeling and approximate inference. The experimental setup is described in Section 4, and results are discussed in Section 5.

2. Related Work

We first present related work of UQ with focus on spatio-temporal reasoning in Section 2.1. Section 2.2 summarizes state-of-the-art results for OnHW recognition.

2.1. UQ for Spatio-Temporal Reasoning

[27] analyzed Bayesian and frequentist UQ methods for spatio-temporal forecasting on network traffic, epidemics and air quality datasets. Their evaluation shows that Bayesian methods are typically more robust in mean prediction, while confidence levels from frequentist methods provide better coverage over data variations (i.e., out-of-distribution data). Furthermore, traditional learning schemes lack knowledge about uncertainty. STU-

aNet [28] addresses this issue for spatio-temporal human mobility forecasting by injecting controllable uncertainty. This allows insights to both, UQ and weak supervised learning. [29] focused on the spatio-temporal uncertainty of urban prediction (where and when a piece of land becomes urban). [7] argue that the feature statistics such as mean and standard deviation (the domain characteristics of the training data), can be manipulated to improve the generalizability of DL models by modeling the uncertainty of domain shifts with feature statistics during training (that follow a multivariate Gaussian distribution). In the context of domain adaptation, [19] addressed the extraction of domain-invariant representations for MTS classification.

2.2. Online Handwriting Recognition

[21] initially proposed the *OnHW-chars* dataset and evaluated machine and DL techniques for the OnHW MTS classification task. The dataset contains right-handed and left-handed writers with a domain shift between both groups of writers (i.e., domains). [25] showed that transfer learning from small adaptation datasets results in poor model performances. Hence, their domain adaptation approach transforms features from left-handed writers into the domain of features from right-handed writers by optimal transport techniques. A reliable UQ method could identify out-of-distribution samples and only apply the transformation on samples for which the model has a high uncertainty. [22] combined offline and online handwriting recognition with a cross-modal representation learning technique by increasing the dataset size by using generative models. A robust uncertainty estimation technique could select samples with high model uncertainty.

3. Methodological Background

In the following we describe Bayesian model averaging in Section 3.1 and the two employed Bayesian UQ methods in Section 3.2. The decomposition of total predictive uncertainty into aleatoric and epistemic uncertainty is discussed in Section 3.3.

3.1. Bayesian Model Averaging

Bayesian approaches in DL naturally represent uncertainty by placing a distribution over model parameters and then marginalizing these parameters to form a predictive distribution (*Bayesian model averaging*) [30]. Let $p(\theta|D)$ be the posterior distribution over model parameters θ , i.e., real-valued weights in the NN, given training dataset D , and let $p(y^*|x^*, \theta)$ denote the probability distribution over model outputs y^* (predicted

¹ Code and datasets: www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html

classes), given sample x^* , and model weights θ . For the OnHW classification task, the sample x^* is an MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_q\} \in \mathbb{R}^{q \times l}$, an ordered sequence of $l = 13$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l}), i \in \{1, \dots, q\}$, where $q = 64$ is the length of the MTS. The training set D is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\} \in \mathbb{R}^{n_U \times q \times l}$, where n_U is the number of time-series. The aim is to predict an unknown class label $y^* \in \mathcal{Y}$ with K classes (i.e., character labels) for a given MTS. The predictive distribution of the target variable is then given by

$$p(y^*|x^*, D) = \int p(y^*|x^*, \theta)p(\theta|D)d\theta. \quad (1)$$

In practice, we can approximate this integral by drawing S Monte Carlo samples from the posterior distribution:

$$p(y^*|x^*, D) \approx \frac{1}{S} \sum_{s=1}^S p(y^*|x^*, \theta_s), \quad \theta_s \sim p(\theta|D). \quad (2)$$

The predicted probability of an outcome is thus a weighted average over its probabilities with the weights being determined by $p(\theta|D)$.

3.2. Approximate Bayesian Inference

In order to apply Bayesian inference to an NN, we need to compute the posterior $p(\theta|D)$ of the NN weights. As the computation of the posterior is usually intractable, a (local) approximation is often used. This can be addressed by SWAG and Deep Ensembles with the latter abstaining from explicitly modeling $p(\theta|D)$ – nevertheless, this method can be considered to be in the field of approximate Bayesian inference.

Stochastic Weight Averaging-Gaussian (SWAG). SWAG [9] is a Bayesian inference technique for DL that builds on Stochastic Weight Averaging (SWA) [31]. SWA computes an average of stochastic gradient decent (SGD) iterates to obtain information about the geometry of $p(\theta|D)$ from its trajectory. This posterior is then approximated by a Gaussian with simplified covariance structure and reduced dimensionality.

Deep Ensembles. Deep Ensembles are a committee of individual NNs initialized with a different seed [12]. The initialization serves as the only source of stochasticity in the model parameters which are otherwise not random; Deep Ensembles can optionally be coupled with a differently shuffled data loader. In contrast to SWAG, results are obtained by averaging the predictions of M independently trained networks instead of explicitly modeling a posterior and sampling from it. [32] point out that even an ensemble size of $M = 5$ performs well, strengthening its reputation as a “gold standard” for accurate and well-calibrated predictive distributions.

3.3. Uncertainty Decomposition

In the literature two sources of uncertainty are commonly considered [13]: (1) *Aleatoric* uncertainty represents stochasticity inherent in the data. For the OnHW application this can be sensor noise induced by the ball-point pen on the paper or by shaky hands of the writer. In particular, even with infinitely many data points, there will always be some variation in the data. (2) *Epistemic* uncertainty is the model uncertainty, which, in theory, can be reduced to zero for an increasing amount of observations. Various approaches of measuring uncertainty exist in the literature. We consider two approaches, both providing justified and mutually complementing insights into our trained models and data situation: uncertainty decomposition based on the softmax output variability [15] in Section 3.3.1 and based on information theory in Section 3.3.2.

3.3.1. Uncertainty Decomposition based on [Kwon et al.]

The definition proposed by [15] is based on considerations by [14] and presents a novel way to estimate predictive uncertainty by breaking it down into

$$\underbrace{\frac{1}{T} \sum_{t=1}^T \text{diag}(\hat{c}_t) - \hat{c}_t \hat{c}_t^\top}_{\text{aleatoric uncertainty}} + \underbrace{\frac{1}{T} \sum_{t=1}^T (\hat{c}_t - \bar{c})(\hat{c}_t - \bar{c})^\top}_{\text{epistemic uncertainty}}, \quad (3)$$

with $\hat{c}_t = (\hat{c}_{t,1}, \dots, \hat{c}_{t,K}) \in [0, 1]^K$ being the softmax output of the NN based on one forward pass (out of T stochastic forward passes), $\sum_{i=1}^K \hat{c}_{t,i} = 1$, and $\bar{c} = \frac{1}{T} \sum_{t=1}^T \hat{c}_t$.

Interpretation. Equation 3 yields two $K \times K$ matrices with different interpretations. For the *aleatoric* part, diagonal values are in $\{x - x^2 \mid x \in [0, 1]\}$, with the maximum uncertainty for $x = 0.5$. If the model is “unsure”, meaning that the model neither displays confidence that a prediction corresponds to a certain class nor displays confidence that it is not, we expect high aleatoric uncertainty. The off-diagonal elements consist of values in $\{-x \cdot y \mid x, y \in [0, 1]\}$, which yields values on the interval $[-0.25, 0]$. Lower values correspond to higher data uncertainty. For the *epistemic* part, the diagonal contains the squared difference to the mean softmax outputs (over T samples). The off-diagonal has positive values when the softmax values coincide and negative values if the softmax values display an inverse relationship.

3.3.2. Uncertainty Decomposition based on Information Theory

Another way to decompose predictive uncertainty into an aleatoric and epistemic part is by following [17] and similarly [16]. Based on principles from information theory, the Shannon entropy $H(p) = -\sum_{i=1}^K p_i \log_2(p_i)$ is utilized as a common measure of “informedness” of a single probability distribution p with K outcomes/classes and the associated probabilities for each i -th class p_i ; taking the logarithm to base 2 yields values measured in *bits*. The total predictive uncertainty (TU) of the predictive distribution $p(y^*|x, D)$ can then be quantified by

$$TU = H(p(y^*|x^*, D)) \approx H\left(\frac{1}{S} \sum_{s=1}^S p(y^*|x^*, \theta_s)\right). \quad (4)$$

Effectively, this is the entropy of the averaged categorical predictions, and it includes the two sources of uncertainty we are interested in.

Aleatoric Uncertainty (AU), Entropy. We can express aleatoric uncertainty as the expectation over the entropies of S sampled conditional predictive distributions with fixed weights, i.e.,

$$AU \approx \frac{1}{S} \sum_{s=1}^S H(p(y^*|x^*, \theta_s)). \quad (5)$$

Epistemic Uncertainty (EU), Mutual Information. Finally, epistemic uncertainty emerges as the difference of total and aleatoric uncertainty $EU = TU - AU$, and is equivalent to the mutual information (MI):

$$EU = H\left(\frac{1}{S} \sum_{s=1}^S p(y^*|x^*, \theta_s)\right) - \frac{1}{S} \sum_{s=1}^S H(p(y^*|x^*, \theta_s)). \quad (6)$$

Intuitively, epistemic uncertainty stands for the information gain about the model parameters that would be obtained when observing the true outcome. MI is always non-negative, zero in case of perfect independence of y^* and θ , and positive when model uncertainty is present at prediction time.

4. Experiments

In our order to evaluate the efficacy of UQ methods for spatio-temporal handwriting datasets, we use the OnHW dataset (Section 4.1) and fit different network architectures (Section 4.2). Our evaluation approach is given in Section 4.3. For architecture and training details and SWAG parameters, see Appendix A.1. For Deep Ensembles, we choose $M = 10$ (for a study on number of base learners in Deep Ensembles vs. SWAG performance, see [9]).

4.1. Online Handwriting Recognition

The *OnHW-chars* [21] dataset consists of recordings from a sensor-enhanced ballpoint pen providing 14 sensor measurements: two accelerometers (3 axes each), one gyroscope (3 axes), one magnetometer (3 axes), a force sensor (with which the pen tip touches the surface), and the time steps. 119 right-handed and nine left-handed writers participated in the data collection. Each person was instructed to write the English alphabet on plain paper six times. This results in 31,275 right-handed and 2,270 left-handed samples. The task is to either classify lowercase letters (26 classes), uppercase letters (26 classes) or combined letters from all 52 classes. For model evaluation, five cross-validation sets are provided by [21] for both writer-dependent (WD) and writer-independent (WI) MTS classification tasks.

4.2. Neural Network Architectures

We use a modified CNN from [21, 24] for feature extraction and combine it with one unit for spatio-temporal classification to extract important temporal features. This unit is added before the last dense layer. We compare a standard long short-term memory (LSTM) cell with 100 neurons, a bidirectional LSTM (BiLSTM) cell with 100 neurons, and a temporal convolutional network (TCN) with 120 neurons. The last dense layer contains 26 neurons for the lowercase and uppercase tasks, or 52 neurons for the combined task. We interpolate the time-series to 64 time steps without sensor normalization.

4.3. Evaluation Metrics

Confidence Calibration. Calibration can be understood as the degree of reliability of a model. According to [18], a predictor is well-calibrated if the derived predictive confidence represents a good approximation of the actual probability of correctness – meaning that 20% of all predictions with a predictive confidence of 80% should actually be false. Calibration is thus a notion of uncertainty, measuring the discrepancy between the model’s forecasts and (empirical) long-run frequencies [12]. Using the definitions of confidence and accuracy [26], we can make statements about over- and under-confidence of the model. We have

$$\text{confidence}(b_e) = \frac{1}{|b_e|} \sum_{s \in b_e} \hat{c}_s \quad (7)$$

and

$$\text{accuracy}(b_e) = \frac{1}{|b_e|} \sum_{s \in b_e} \mathbb{1}(\hat{y}_s = y_s), \quad (8)$$

with b_e denoting the set of indices of sampled softmax outputs falling into the interval $(l_e, u_e]$. Commonly, the softmax output range is divided into ten bins

Method		Lowercase		Uppercase		Combined	
		WD	WI	WD	WI	WD	WI
SWAG	right	83.73	76.27	87.10	81.69	72.13	65.41
	left	55.51	45.91	55.04	50.67	46.08	39.26
Deep Ensembles	right	83.07	73.87	89.92	80.86	75.29	64.22
	left	45.25	37.00	62.73	48.31	45.95	33.27
Best BNN Method (right-handed)	right	84.44	76.96	90.31	82.21	75.51	66.12
	left	42.55	44.19	49.87	48.54	33.68	36.20

Table 1

Accuracies (in %) for best models trained on right- and left-handed data and evaluated on *right-handed or left-handed* writers data *separately*, compared to the best performing models which were only trained on right-handed data. **Bold**: best results.

Method	Lowercase		Uppercase		Combined	
	WD	WI	WD	WI	WD	WI
Frequentist [21]	84.62	76.85	89.89	83.01	70.50	64.13
	TCN	TCN	TCN	TCN	TCN	LSTM
SWAG	84.44	76.96	87.58	82.21	72.54	66.12
	TCN	TCN	TCN	TCN	TCN	TCN
Deep Ensembles	83.43	73.41	90.31	81.26	75.51	64.21
	BiLSTM	TCN	TCN	TCN	TCN	TCN

Table 2

Accuracies (in %) for models trained on *right-handed* writers data and evaluated on *right-handed* writers data. Second row shows the respective model. **Bold**: best results.

(interval sizes of 0.1). We can now make statements whether our model is under-confident ($\text{accuracy}(b_e) > \text{confidence}(b_e)$) or over-confident ($\text{accuracy}(b_e) < \text{confidence}(b_e)$). It has been shown that softmax outputs of deep NNs are in general not well calibrated and are often either over- or under-confident [26]. Ideally, $\text{accuracy}(b_e) \approx \text{confidence}(b_e)$, allowing the user to interpret softmax outputs as probabilities and thereby quantify the prediction uncertainty.

Expected Calibration Error (ECE). The ECE summarizes how far away the confidence is from the actual (empirical) accuracy [26]. It can be defined as

$$\text{ECE}(b_e) = \sum_{e=1}^E \frac{|b_e|}{n} |\text{accuracy}(b_e) - \text{confidence}(b_e)|, \quad (9)$$

with n being the number of predicted softmax outputs, and E being the number of bins. Note that this metric does not give any information about over- or under-confidence – only how far away the expected accuracy is from the confidence. Ideally, the ECE is 0.

Reliability Diagrams. We visualize Equations 7 and 8 as reliability diagrams [33] for selected models. Generally, a model is over-confident if the black bars (displaying the accuracy for one bin) are below the dashed bisectors. Consequently, if the black bars are above the bisectors,

Method	Lowercase		Uppercase		Combined	
	WD	WI	WD	WI	WD	WI
SWAG	81.85	74.24	84.92	79.58	70.37	63.64
	TCN	TCN	TCN	TCN	TCN	TCN
Deep Ensembles	80.55	71.41	88.07	78.65	73.31	62.14
	LSTM	TCN	TCN	TCN	TCN	TCN

Table 3

Accuracies (in %) for models trained on *right- and left-handed* writers data and evaluated on *right-handed* writers data. Second row shows the respective model. **Bold**: best results.

the model is under-confident. We additionally plot the histogram [34] of the softmax outputs to get an overview of the distribution.

5. Experimental Results

In the following, we summarize the main results. In general, the models perform better on WD classification tasks than on WI tasks. Architectures with TCN units outperform LSTM and BiLSTM units on most tasks.

Evaluation on Handedness (trained on right-handed writers). SWAG and Deep Ensemble models perform very similarly to frequentist models proposed in [21] in terms of predictive accuracy (see Table 2), being at most 3% points below and 5% points above a respective frequentist model. When applying models trained with right-handed data on the left-handed datasets, the performance ranges from 33.27% to 49.87% accuracy (see Table 1) which is substantially better than “pure guessing” – our models make informed decisions after shifting domains, albeit at a lower standard. A possible reason is that certain sensors produce nearly identical signals regardless of the orientation of the pen. For example, the accelerometer at the bottom of the pen should give the same readings for left-handed writers when writing “I” and “i” as for right-handed writers, since it is simply a downward motion regardless of the writing hand.

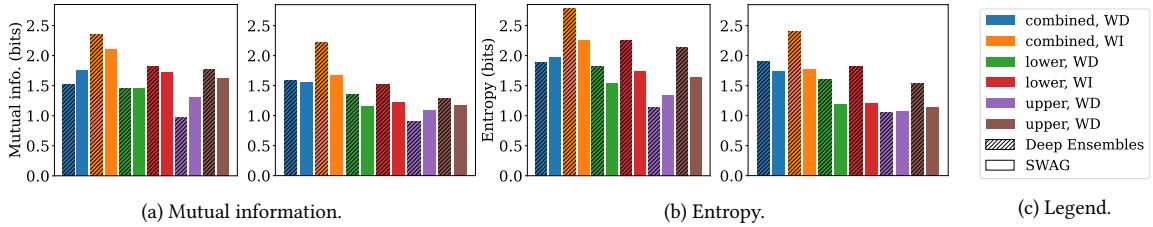


Figure 1: Information theoretic uncertainty measures for the Deep Ensemble (dashed) and SWAG (non-dashed) CNN+TCN models. The models are trained on the combined right- and left-handed writers datasets (a and b, left) or only the right-handed writers dataset (a and b, right), and evaluated on the left-handed writers dataset. We provide results for lowercase, uppercase and combined classification tasks.

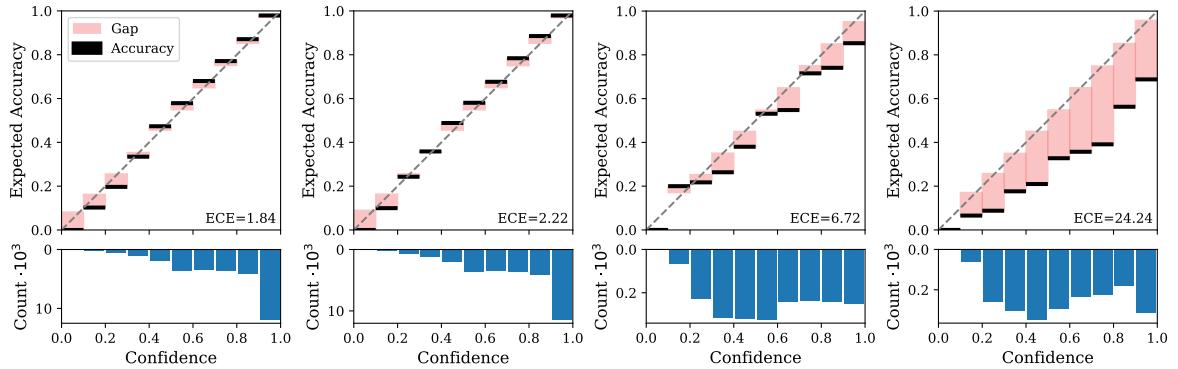


Figure 2: Reliability diagram for the Deep Ensemble CNN+TCN model trained on the combined WD datasets. a) and c): Trained on the combined right- and left-handed writers datasets. b) and d): Trained on right-handed writers only.

Evaluation on Handedness (trained on right- and left-handed writers). When evaluating performance on right-handed data, models trained only on right-handed datasets consistently outperform models trained on both datasets combined and yield between 2% points and 12% points higher accuracies (see Table 3). This performance loss is compensated by a performance gain for left-handed data. Still, the performance is not up to par with right-handed data; this gap may be due to a “writing style” particular to every writer that especially influences the gyroscope and magnetometer measurements. More importantly, left-handed writers have a writing style different to right-handed writers which is perhaps exactly what the right-handed models never learned in order to address the style of left-handed writers, underlining the need for a sufficient amount of samples to get a good representation of various writing styles.

Analysis of Uncertainty. Figure 1 shows the MI and entropy for SWAG and Deep Ensemble models evaluated on the left-handed data. The barplots show that the models trained on only right-handed data display lower uncertainty (i.e., higher confidence) compared to models trained on combined data. However, this higher

confidence is not empirically justified when looking at the reliability diagrams in Figure 2, which point out that models trained without left-handed writers data are miscalibrated and therefore overconfident. Models trained on the combined writers (Figures 2a and 2c) provide more realistic accuracies when applied to the left-handed data (ECE of 6.72). The ECE is even higher (24.24) for left-handed evaluation without left-handers in the training set (see Figure 2d). For a separate evaluation for each character, see Appendix A.2.

5.1. Uncertainty Analysis based on [Kwon et al.]

In Figure 3 we visualize the aleatoric and epistemic uncertainty as well as the confusion matrix for the Deep Ensemble model and the combined task. For SWAG model results, see Appendix A.3. In the aleatoric uncertainty heatmap (Figure 3a) we observe a trace with negative values at the lower end of the scale. Note that for off-diagonal values, the aleatoric uncertainty is higher for lower softmax values. Here, two softmax outputs (with the highest values) coincide on average (see Section 3.3.1). This means that the model tends to confuse the two

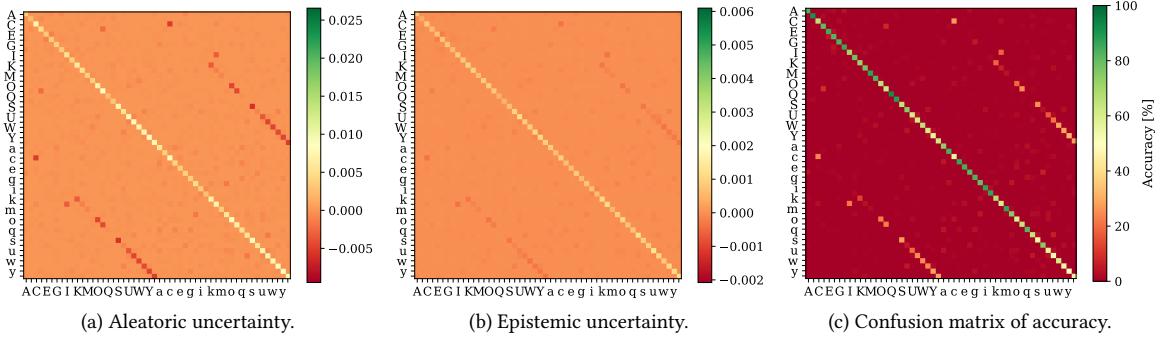


Figure 3: Uncertainty prediction for the Deep Ensemble CNN+TCN model trained on the combined WD (right-handed only) dataset. Note that the color scale is fixed for all subplots for comparability with Figure 4, and that we skipped every second character label for readability.

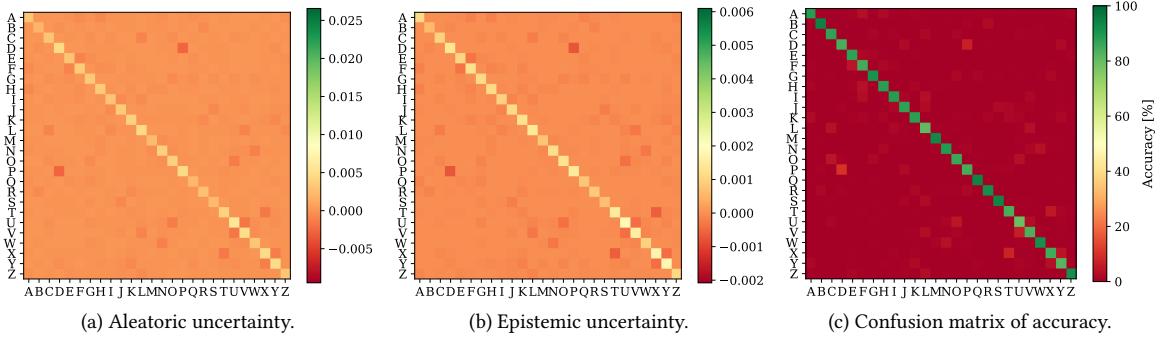


Figure 4: Uncertainty prediction for the Deep Ensemble CNN+TCN model trained on the uppercase WD (right-handed only) dataset. Note that the color scale is fixed for all subplots for comparability with Figure 3 and 6.

classes. The most prominent off-diagonal strip corresponds to the upper- and lowercase pairs. This makes intuitive sense since, e.g., the lowercase "u" and uppercase "U" are written similarly. This effect is consequently not present for less similar pairs like "a" and "A". We can see this effect also for "l" (lowercase "L") and "I" (uppercase "i"). A very similar pattern can also be observed in the confusion matrix (see Figure 3c), confirming that the trained model is not only unsure about how to classify these pairs, but is also empirically worse in the respective classification task.

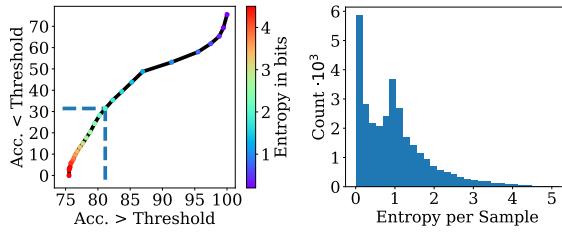
These patterns allow for further interesting insights. For example, one might expect this pattern to occur for "i" and "j", but the corresponding heatmap entries lack signs of confusion of the model. Similarity between characters consequently hinges on the similarity of *motions* while writing. Two characters with small differences are written similarly but in different *sizes*. This also holds for specific parts of the letters. For example, "n" and "h" have a higher aleatoric uncertainty in Figure 3a; the major difference being that one tiny part of "h" is longer.

Somewhat puzzling is that we see the same effect in the epistemic uncertainty heatmap (see Figure 3b), where

such pairs with high similarity lead to negative values. When one entry of the softmax output values is below and another entry above the respective sample mean, negative epistemic uncertainty is implied. This leads to some kind of discriminative power due to the negative "covariance" for which there is little justification. We thus advise caution when interpreting the epistemic uncertainty in this context.

5.2. Uncertainty based on Information Theory

We further highlight the trade-off when using information theory-based measures to decide whether a sample is too uncertain to classify correctly. This is depicted by Figure 5a showing the relationship between classification accuracies and different threshold values. We choose the entropy as the target metric for uncertainty evaluation (MI would work analogously). On the x-axis is the accuracy of the samples above the threshold, i.e., samples our model feels confident about classifying correctly. On the y-axis is the accuracy for the samples below the threshold. These values would be considered as too inaccurate



(a) Sample accuracies below and (b) Histogram visualizing the above an entropy threshold. entropy distribution.

Figure 5: Accuracy and entropy for the Deep Ensemble CNN+TCN model trained on the combined WD (right-handed only) dataset.

to confidently classify. Setting the threshold to 2.0 bits would approximately yield an accuracy of 82% for the observations above this threshold and approx. 31% accuracy for observations below this threshold (emphasized by the dashed lines). Figure 5b depicts the entropy distribution and further clarifies this point. Convincingly, the accuracy reduces to almost zero for very high entropy samples. Note that the accuracy does not need to decrease with an increasing entropy threshold or even be zero for very high entropy values, even though this is generally true for our models.

5.3. Summary and Limitations

Uncertainty Decomposition. Neither uncertainty quantification method shows notable differences between aleatoric and epistemic uncertainty. The heatmaps exhibit the same “strip” for similar characters and give no hints to different sources of uncertainty (data-driven or systemic confusion). The benefits of this kind of uncertainty differentiation are limited, but measuring the total uncertainty can still be useful for domain adaptation or the detection of wrong labels.

Real-World Link. Since the models trained on right- and left-handed writers lead to lower data confidence compared to models trained only on right-handed writers (see Figure 1), it is unclear how well the measured MI and entropy translate to the real-world uncertainty. Therefore, verifying uncertainty remains a limitation in our interpretation. While we can discriminate between the entropy associated with different samples, pre-defining thresholds for uncertain samples is challenging due to the following reasons: (1) Raw sensor data is elaborate to interpret and making statements about, e.g., the writing style from sensor data is hardly possible – which, in turn, is connected to model uncertainty. (2) Interpreting the *graphomotoricity* qualitatively, e.g., for teaching hand writing, a qualified expert in this field is required. (3) Different writing domains (different pens, surfaces etc.) lead to different requirements for the uncertainty threshold.

6. Conclusion

We employed SWAG and Deep Ensembles for OnHW recognition with left- and right-handed writers, a spatio-temporal MTS classification task with domain shift. We critically evaluated aleatoric and epistemic uncertainty using confidence calibration, ECE and reliability diagrams. In summary, (1) the model performance only partly relates to the handedness of writers, (2) our models are over-confident and miscalibrated when only trained with right-handed writers and evaluated on left-handed writers, (3) the uncertainty of the models for small and capital characters combined is related to lower classification accuracy, and (4) the entropy and mutual information for individual samples correlate well with the accuracy of our models. Our comparison of different ways to decompose uncertainty easily generalizes to other classification tasks and can be useful for spatio-temporal reasoning. In terms of Bayesian inference, SWAG and Deep Ensemble models perform similarly, while SWAG is computationally less expensive.

Acknowledgments

This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A (David Rügamer) and by the research program Human-Computer-Interaction through the project “Schreibtrainer”, Grant No. 16SV8228, as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

References

- [1] B. Sun, J. Feng, K. Saenko, Correlation Alignment for Unsupervised Domain Adaptation, in: *arXiv:1612.01939*, 2016.
- [2] B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, Y. Bengio, Toward Causal Representation Learning, in: *Proceedings of the IEEE*, volume 109(5), 2021, pp. 612–634. doi:10.1109/JPROC.2021.3058954.
- [3] S. J. Pan, Q. Yang, A Survey on Transfer Learning, in: *Trans. on Knowledge and Data Engineering*, volume 22(10), 2009, pp. 1345–1359. doi:10.1109/TKDE.2009.191.
- [4] L. Shao, F. Zhu, X. Li, Transfer Learning for Visual Categorization: A Survey, in: *Trans. on Neural Networks and Learning Systems*, volume 26(5), 2014, pp. 1019–1034.
- [5] M. Long, J. Wang, G. Ding, J. Sun, P. S. Yu, Transfer

Joint Matching for Unsupervised Domain Adaptation, in: *CVPR*, Columbus, OH, 2014, pp. 1410–1417.

[6] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting Visual Category Models to New Domains, in: *ECCV*, volume 6314, 2010, pp. 213–226.

[7] X. Li, Y. Dai, Y. Ge, J. Liu, Y. Shan, L. Duan, Uncertainty Modeling for Out-of-Distribution Generalization, in: *ICLR*, 2022.

[8] M. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, A. Srivastava, Fast and Scalable Bayesian Deep Learning by Weight-Perturbation in Adam, in: *JMLR*, volume 80, 2018, pp. 2611–2620.

[9] W. J. Maddox, T. Garipov, P. Izmailov, D. Vetrov, A. G. Wilson, A Simple Baseline for Bayesian Uncertainty in Deep Learning, in: *NIPS*, 2019, pp. 13153–13164.

[10] C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight Uncertainty in Neural Network, in: *ICML*, volume 37, 2015, pp. 1613–1622.

[11] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, P. Hennig, Laplace Redux - Effortless Bayesian Deep Learning, in: *NIPS*, 2021.

[12] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles, in: *NIPS*, 2017, pp. 6405–6416.

[13] E. Hüllermeier, W. Waegeman, Aleatoric and Epistemic Uncertainty in Machine Learning: An Introduction to Concepts and Methods, in: *Machine Learning*, volume 110, 2021, pp. 457–506.

[14] A. Kendall, Y. Gal, What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision, in: *NIPS*, volume 30, 2017, pp. 5580–5590.

[15] Y. Kwon, J.-H. Won, B. J. Kim, M. C. Paik, Uncertainty Quantification Using Bayesian Neural Networks in Classification: Application to Ischemic Stroke Lesion Segmentation, in: *ICLR*, 2018.

[16] L. Smith, Y. Gal, Understanding Measures of Uncertainty for Adversarial Example Detection, in: *UAI*, 2018.

[17] S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, S. Udluft, Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-sensitive Learning, in: *JMLR*, volume 80, 2018, pp. 1184–1193.

[18] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, M. Shahzad, W. Yang, R. Bamler, X. X. Zhu, A Survey of Uncertainty in Deep Neural Networks, in: *arXiv:2107.03342*, 2021.

[19] R. Cai, J. Chen, Z. Li, W. Chen, K. Zhang, J. Ye, Z. Li, X. Yang, Z. Zhang, Time Series Domain Adaptation via Sparse Associative Structure Alignment, in: *AAAI*, volume 216, 2014, pp. 76–102.

[20] R. Plamondon, S. N. Srihari, On-line and Off-line Handwriting Recognition: A Comprehensive Survey, in: *TPAMI*, volume 22(1), 2000, pp. 63–84.

[21] F. Ott, M. Wehbi, T. Hamann, J. Barth, B. Eskofier, C. Mutschler, The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning, in: *IMWUT*, volume 4(3), article 92, Cancún, Mexico, 2020.

[22] F. Ott, D. Rügamer, L. Heublein, B. Bischl, C. Mutschler, Cross-Modal Common Representation Learning with Triplet Loss Functions, in: *arXiv:2202.07901*, 2022.

[23] F. Ott, D. Rügamer, L. Heublein, B. Bischl, C. Mutschler, Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach, in: *WACV*, Waikoloa, HI, 2022, pp. 266–276.

[24] F. Ott, D. Rügamer, L. Heublein, T. Hamann, J. Barth, B. Bischl, C. Mutschler, Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens, in: *arXiv:2202.07036*, 2022.

[25] F. Ott, D. Rügamer, L. Heublein, B. Bischl, C. Mutschler, Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift, in: *arXiv:2204.03342*, 2022.

[26] C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger, On Calibration of Modern Neural Networks, in: *ICML*, volume 70, 2017, pp. 1321–1330.

[27] D. Wu, L. Gao, X. Xiong, M. Chinazzi, A. Vespignani, Y.-A. Ma, R. Yu, Quantifying Uncertainty in Deep Spatiotemporal Forecasting, in: *arXiv:2105.11982*, 2021.

[28] Z. Zhou, Y. Wang, X. Xie, L. Qiao, Y. Li, STUANet: Understanding Uncertainty in Spatiotemporal Collective Human Mobility, in: *WWW*, 2021.

[29] J. A. Gómez, C. Guan, P. Tripathy, J. C. Duque, S. Pasos, M. Keith, J. Liu, Analyzing the Spatiotemporal Uncertainty in Urbanization Predictions, in: *Remote Sensing*, volume 13(512), 2021.

[30] J. A. Hoeting, D. Madigan, A. E. Raftery, C. T. Volinsky, Bayesian Model Averaging: A Tutorial, in: *Statist. Sci.*, volume 14(4), 1999, pp. 382–417.

[31] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, A. G. Wilson, Averaging Weights Leads to Wider Optima and Better Generalization, in: *UAI*, 2018.

[32] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, J. Snoek, Can You Trust Your Model’s Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift, in: *NIPS*, volume 32, 2019, pp. 14003–14014.

[33] M. H. Degroot, S. E. Fienberg, The Comparison and Evaluation of Forecasters, in: *The Statistician*, volume 32, 1983.

[34] M. Hollemans, Reliability Diagrams, <https://github.com/hollance/reliability-diagrams>, 2020.

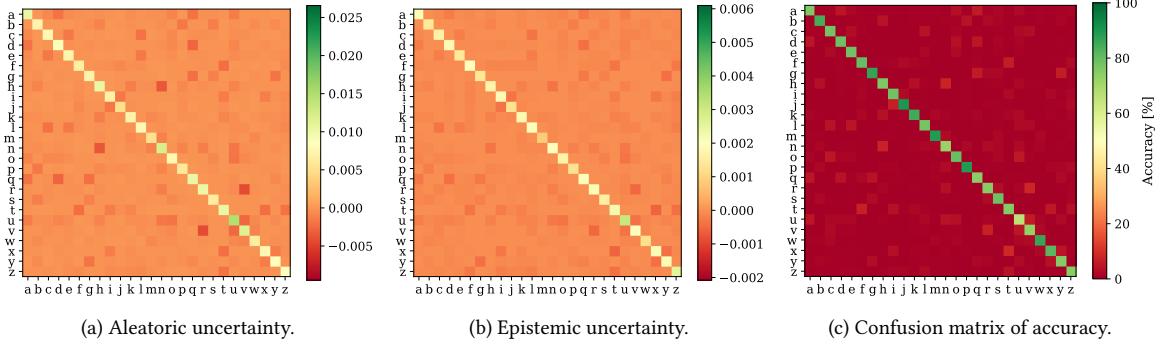


Figure 6: Uncertainty prediction for the Deep Ensemble CNN+BiLSTM model (which outperformed the TCN-based architecture) trained on the lowercase WD (right-handed only) dataset. Note that the color scale is fixed for all subplots for comparability with Figure 3 and 4.

A. Appendices

We propose model parameters in Section A.1 and show an evaluation per character in Section A.2. We propose results for the SWAG model in Section A.3.

A.1. Model and UQ Method Parameters

For reproducibility, we state all general model architecture parameters and propose training parameters for the SWAG model. For all experiments we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM coupled with Intel Core Xeon CPUs and 192 GB RAM.

Model Parameters. We use a CNN with dropout rate 20%, convolutional layers with kernel size 4 and filter size 200. The temporal cell (LSTM, BiLSTM or TCN) contains 100, 100 or 120 neurons, respectively. We interpolate the time-series to 64 time steps, and train the model for 2,000 epochs with early stopping and a batch size of 50.

SWAG Parameters. We initialize the stochastic gradient descent (SGD) optimizer with initial learning rate 10^{-2} , a momentum of 0.9, and weight decay of 10^{-4} . The stochastic weight averaging (SWA) burn-in period was run for 10 epochs. SWAG showed a training process with fast convergence.

A.2. Evaluation per Character

Confusion Matrices. We propose the confusion matrices for the aleatoric and epistemic uncertainty as well as the accuracy (in %) for the uppercase (see Figure 4) and lowercase (see Figure 6) datasets. While for the combined training, lower- and uppercase characters are often misclassified, the separate training leads to confusion of characters with similar shapes, e.g., for the uppercase task, the model is uncertain for "D" and "P", "U" and

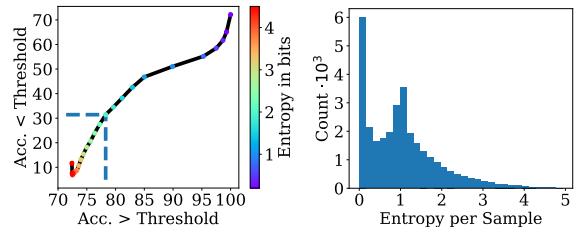


Figure 7: Accuracy and entropy for the SWAG CNN+TCN model trained on the combined WD (right-handed only) dataset.

"V", and "T" and "X". These confusions can be identified with the aleatoric and epistemic uncertainty and correspond with the classification accuracies. Overall, the uncertainty for lowercase characters is higher (see Figure 6a) since the writing style of lowercase characters is oftentimes quite similar, e.g., "r" and "v", "u" and "v", "h" and "n", and "d" and "q". This also leads to a lower classification accuracy (see Figure 6c).

Mutual Information and Entropy. Figure 8a shows the mutual information (MI) per character, and Figure 8b shows the entropy, respectively. In general, the MI and entropy correlates and are similar for each character. The MI and entropy is high for the characters "U", "u", "v", "x", and "z". Furthermore, both metrics are higher for lowercase characters than for uppercase characters. This corresponds to the confusion matrices in Figure 4 and 6 where aleatoric uncertainty is higher for off-diagonals for lowercase characters.

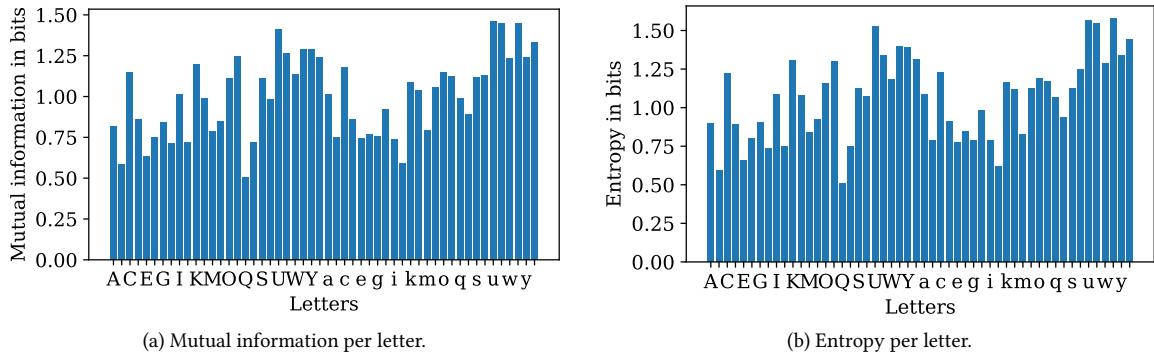


Figure 8: Mutual information and entropy per letter for the Deep Ensemble CNN+TCN model trained on the combined WD (right-handed only) dataset. Note that we skip every second character in the x-axis (ordered alphabetically) for readability.

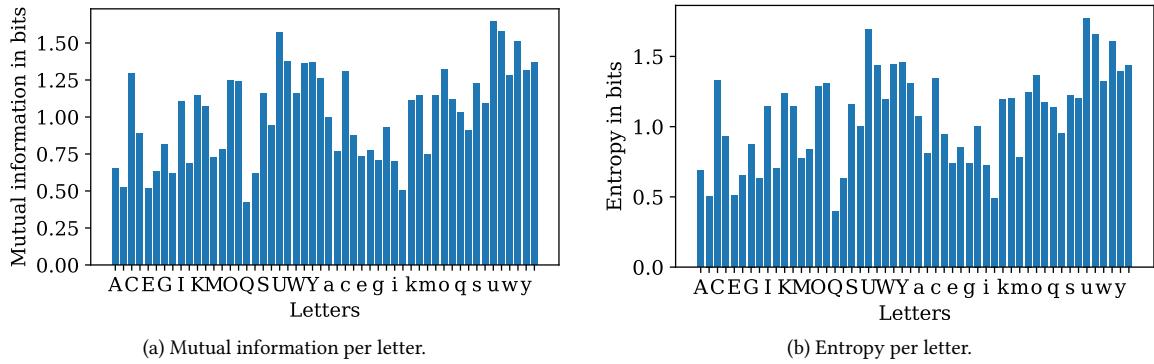


Figure 9: Mutual information and entropy per letter for the SWAG CNN+TCN model trained on the combined WD (right-handed only) dataset. Note that we skip every second character in the x-axis (ordered alphabetically) for readability.

A.3. SWAG Model Results

This section provides plots for the SWAG model that can directly be compared to the previously shown Deep Ensemble model plots. We observe very similar results

between SWAG and Deep Ensemble models. Figure 9 shows the MI and entropy for the SWAG model with the same pattern as for the Deep Ensemble model with lower absolute values. In Figure 10, we see the same overconfi-

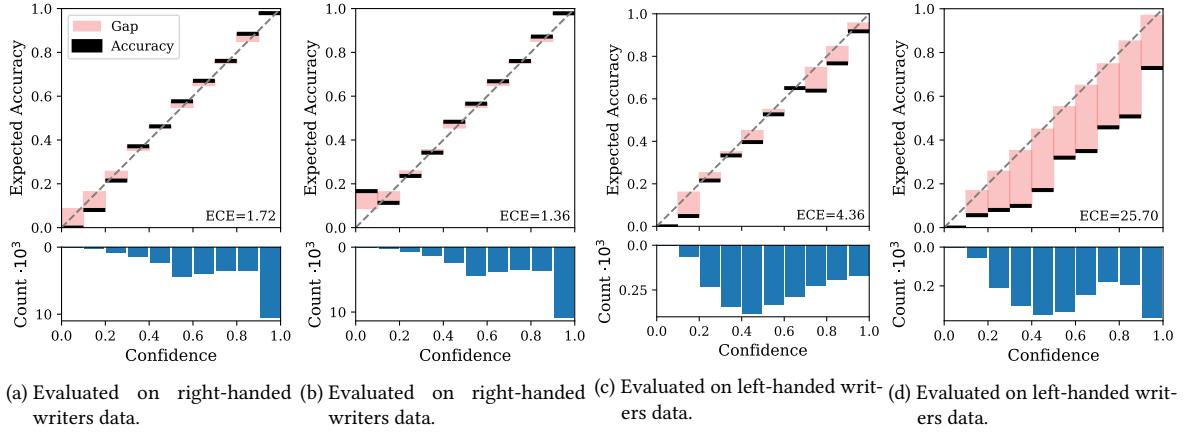


Figure 10: Reliability diagram for the SWAG CNN+TCN model trained on the combined WD datasets. a) and c): Trained on the combined right- and left-handed writers datasets. b) and d): Trained on right-handed writers only.

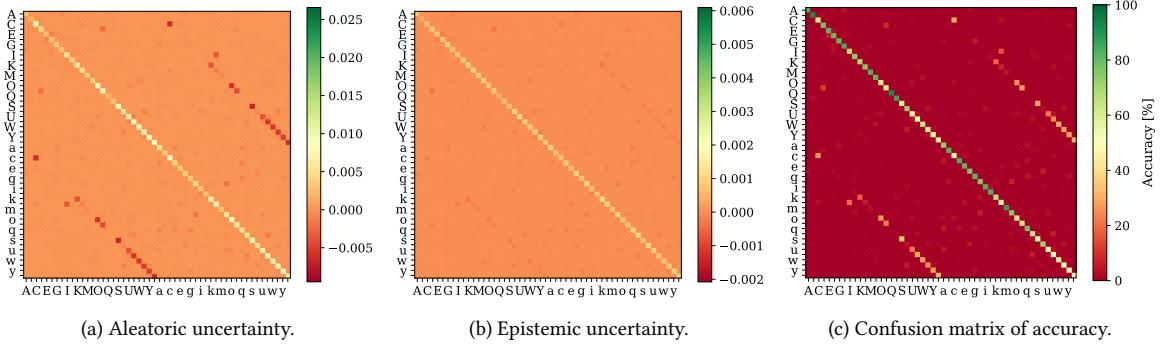


Figure 11: Uncertainty prediction for the SWAG CNN+TCN model trained on the combined WD (right-handed only) dataset. Note that the color scale is fixed for all subplots for comparability with the other heatmaps, and that we skipped every second character label for readability.

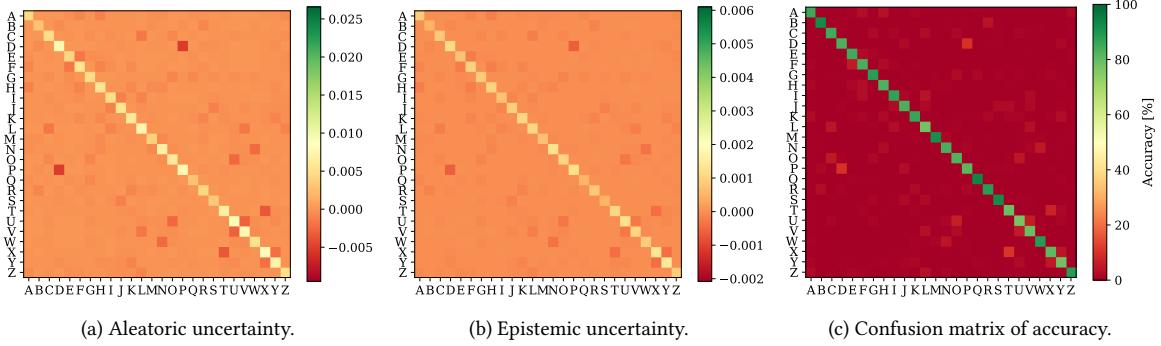


Figure 12: Uncertainty prediction for the SWAG CNN+TCN model trained on the uppercase WD (right-handed only) dataset. Note that the color scale is fixed for all subplots for comparability with the other heatmaps.

dence on left-handed data for SWAG models that have never seen this data similar as for Deep Ensemble models. The ECE by the SWAG model is marginally lower than the ECE by the Deep Ensemble model, but follows the same trend. The heatmaps in Figures 11 for lowercase and uppercase characters, in Figure 12 for uppercase

characters only, and in Figure 13 for lowercase characters only of the SWAG model show the same pattern as the heatmaps for Deep Ensemble models.

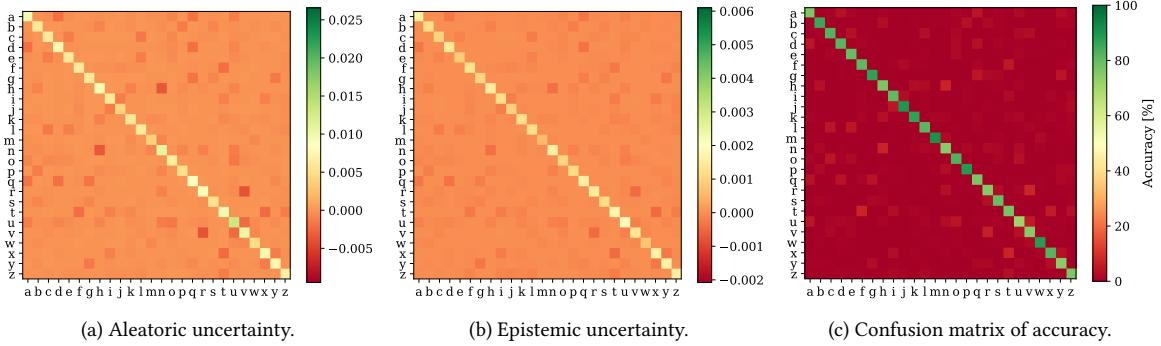


Figure 13: Uncertainty prediction for the SWAG CNN+TCN model trained on the lowercase WD (right-handed only) dataset. Note that the color scale is fixed for all subplots for comparability with the other heatmaps.

Chapter 6

Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens

Contributing Article

Felix Ott, David Rügamer, Lucas Heublein, Tim Hamann, Jens Barth, Bernd Bischl, and Christopher Mutschler. Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In *International Journal on Document Analysis and Recognition (IJDAR)*, volume 25(12), pages 385–414, September 2022. doi:10.1007/s10032-022-00415-6.

Publisher Website

<https://link.springer.com/article/10.1007/s10032-022-00415-6>

Copyright License

This work is licensed under a Creative Commons Attribution International 4.0 License (<https://creativecommons.org/licenses/by/4.0/>). © Copyright held by the owner/author(s).

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing,

visualization, and project administration. David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Lucas Heublein contributed for the methodology (i.e., creation of models), software, validation, investigation, and data curation (i.e., data recording and data management). Bernd Bischl contributed by supervision. Tim Hamann and Jens Barth contributed with the investigation (i.e., data collection and data management), resources (i.e., providing of sensor-enhanced pens for data collection), project administration, reviewing and editing, and provided images for Figure 2. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), visualization (i.e., presentation of the published work), supervision, and funding acquisition of the project “Schreibtrainer” (grant number 16SV8228) by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”. This paper was presented by Felix Ott at the International Conference on Frontiers in Handwriting Recognition (ICFHR) in December 2022.

Datasets

This paper uses the publicly available IAM-OffDB, VNOnDB, and OnHW-chars datasets. Along this paper, the OnHW-words500, OnHW-wordsTraj, OnHW-symbols, and split OnHW-equations datasets were published and are available at:

<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>

Statement about Recent, Related Research¹⁷

The paper by Ghosh et al. (2022) provides a recent survey of OnHW datasets, including a detailed comparison of the OnHW-chars dataset with 13 other OnHW datasets. In another work, Wegmeth et al. (2021) employed the OnHW-equations dataset published in the contributing paper (Ott et al., 2022d) and developed a method for classifying mathematical equations. Their approach involves a data-dependant and rule-based information extraction algorithm for equations segmentation, and a combination of a binary random forest for classifying the writing activity and a CNN for making single-label predictions on windows of data. Furthermore, Bronkhorst (2021) demonstrated the effectiveness of Transformers, but only evaluated their approach on single characters using the OnHW-chars dataset.

The contributing paper (Ott et al., 2022d) employed various state-of-the-art time-series classification techniques implemented in the tsai toolbox (Oguiza, 2020) to conduct a benchmark on OnHW datasets. Recently, the tsai toolbox has been updated to include the implementation of gating multi-layer perceptron (gMLP) proposed by Liu et al. (2021). This method outperforms Transformers in vision applications and may be of interest for future time-series applications. We explored alternative loss functions to the categorical cross-entropy (CE) such as the focal loss, label smoothing, both soft and hard, the generalized and symmetric categorical CE, and joint optimization on the single characters

OnHW datasets. Additionally, the *label relaxation* technique by Lienen & Hüllermeier (2021) can be applied on the OnHW datasets, where the target is a set of probabilities represented as an upper probability distribution. The *label relaxation* method provides a suitable closed-form expression for model optimization based on the Kullback-Leibler divergence.

Gu et al. (2022) employed the convolutional RNN proposed in the contributing paper (Ott et al., 2022d) to extract features from IMU signals for the validation of in-ear headphones movements. The network comprises 1D convolutional layers and bidirectional gated recurrent units, with an eight-channel input representing accelerometer and gyroscope data. This highlights the versatility of the best-performing model CNN+BiLSTM from Ott et al. (2022d) in other applications with distinct data dimensions. Additionally, Teng et al. (2022) utilized a CNN with a BiLSTM layer along with the connectionist temporal classification (CTC) loss for Mongolian OnHW word recognition. The CNN+BiLSTM network has been applied to offline signature verification by Longjam et al. (2023). Signature verification is a challenging task due to the high similarity between different individuals and the variations within individuals, as well as the skilled imitation of signature structure. Therefore, CNN+BiLSTM models, along with Transformer networks, are leading approaches for both offline and online HWR tasks. The IAMOnDB dataset was benchmarked by Mustafid et al. (2022) using various methods based on neural networks, graph neural networks, attention-based neural networks, and Transformers. Building on the OnHW-chars dataset of Ott et al. (2020b) and the benchmark of Ott et al. (2022d), Azimi et al. (2022) developed both ML models (i.e., decision tree, random forest, extra trees, logistic regression, kNN, and SVM) and DL models (i.e., FCN, LSTM, BiLSTM, (Bi)LSTM-FCN, M(Bi)LSTM-FCN, ResNet, ResCNN, InceptionTime, XceptionTime, and XCM). These models achieve an improvement of 11.3% to 23.56% through hyperparameter tuning. The best performing model is InceptionTime, which aligns with the results of the contributing paper (Ott et al., 2022d). However, as only one dataset was evaluated, it is not possible to make a general statement about the performance. The improvement of 3.08% to 7.01% due to ensemble learning with DL models is relevant and can be applied to all OnHW datasets in future studies.

In Section 6.1 of the contributing paper (Ott et al., 2022d), we presented an argument for the social impact of OnHW recognition for writing on normal paper, emphasizing that the visual feedback provided by the pen can assist young students and children in learning a new language. The aim of OnHW with sensor-enhanced pens is to support self-paced learning from home without additional effort (Drey et al., 2022). Our statements are supported by recent research by Ihara et al. (2021), who found that handwriting on paper is more effective for learning than typing on a keyboard. They conduct a study comparing learning by handwriting with a digital pen on a tablet, typing on a keyboard, or handwriting with an ink pen on paper, and measured behavioral and electroencephalographic indices immediately after learning with each writing tool. Their results show a great priming effect for words learned with the digital or ink pen than those learned with the keyboard, and a greater priming effect for words learned with the ink pen compared with words learned by typing, particularly for an unfamiliar group with respect to the digital pen. In addition, the

study found that positive mood during learning was significantly higher during handwriting than during typing (Ihara et al., 2021), supporting the motivation for the sensor-enhanced pen. The use of writing tools does not influence behavior, but does contribute to greater memorization of new words for handwriting. Furthermore, Wiechmann et al. (2022) split 68 students into longhand, laptop, or tablet note-taking groups, and assessed these students for conceptual and factual recall while taking notes. The authors found no significant difference in the recall scores, but a significantly higher median word count for laptops compared to tablets and handwriting. Students are encouraged to pick the most appealing note-taking method. Therefore, the sensor-enhanced pen can serve as an alternative method for taking notes.

The work of Wu et al. (2023) is relevant in the field of handwriting recognition, albeit using different sensors. Their proposed approach, DMHC, is a multi-modal recognition system that operates without requiring any device and fuses two distinct acoustic modalities (i.e., ultrasonic and audio signals) to achieve noise-resistant performance.



Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU-enhanced pens

Felix Ott^{1,2} · David Rügamer^{2,3} · Lucas Heublein¹ · Tim Hamann⁴ · Jens Barth³ · Bernd Bischl² · Christopher Mutschler¹

Received: 31 March 2022 / Revised: 2 June 2022 / Accepted: 1 September 2022
 © The Author(s) 2022

Abstract

Handwriting is one of the most frequently occurring patterns in everyday life and with it comes challenging applications such as handwriting recognition, writer identification and signature verification. In contrast to offline HWR that only uses spatial information (i.e., images), online HWR uses richer spatio-temporal information (i.e., trajectory data or inertial data). While there exist many offline HWR datasets, there are only little data available for the development of OnHWR methods on paper as it requires hardware-integrated pens. This paper presents data and benchmark models for real-time sequence-to-sequence learning and single character-based recognition. Our data are recorded by a sensor-enhanced ballpoint pen, yielding sensor data streams from triaxial accelerometers, a gyroscope, a magnetometer and a force sensor at 100 Hz. We propose a variety of datasets including equations and words for both the writer-dependent and writer-independent tasks. Our datasets allow a comparison between classical OnHWR on tablets and on paper with sensor-enhanced pens. We provide an evaluation benchmark for seq2seq and single character-based HWR using recurrent and temporal convolutional networks and transformers combined with a connectionist temporal classification (CTC) loss and cross-entropy (CE) losses. Our convolutional network combined with BiLSTMs outperforms transformer-based architectures, is on par with InceptionTime for sequence-based classification tasks and yields better results compared to 28 state-of-the-art techniques. Time-series augmentation methods improve the sequence-based task, and we show that CE variants can improve the single classification task. Our implementations together with the large benchmark of state-of-the-art techniques of novel OnHWR datasets serve as a baseline for future research in the area of OnHWR on paper.

Keywords Online handwriting recognition · Sequence-based and character datasets · Time-series data · Sensor-enhanced pen · Writer-(in)dependent · Architectures

1 Introduction

Handwriting provides language information based on structured symbols and is used for communication or documentation of speech. HWR refers to the digitalization of written text and can be categorized into offline and online HWR. Research on offline HWR systems is very advanced and has almost reached a human-level performance, but cannot be applied for real-time recognition applications (as they induce

Felix Ott
 felix.ott@iis.fraunhofer.de

David Rügamer
 david.ruegamer@stat.uni-muenchen.de

Lucas Heublein
 heublein@iis.fraunhofer.de

Tim Hamann
 tim.hamann@stabilo.com

Jens Barth
 jens.barth@stabilo.com

Bernd Bischl
 bernd.bischl@stat.uni-muenchen.de

Christopher Mutschler
 christopher.mutschler@iis.fraunhofer.de

¹ Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Nürnberg, Germany

² LMU Munich, Munich, Germany

³ RWTH Aachen, Aachen, Germany

⁴ STABILO International GmbH, Heroldsberg, Germany

an unacceptable delay) [23] as the written text has first to be digitalized. Optical character recognition (OCR), one of the dominant approaches in offline HWR, deals with the analysis of the visual representation of handwriting only. Its application and accuracy are limited as it cannot make use of temporal information such as writing direction and speed, or the pressure applied to the paper [65,69].

In contrast, online HWR typically works on different types of spatio-temporal signals such as the positions of the pen tip (in 2D), its temporal context or the movement on the writing surface. These handwriting signals can then, e.g., be partitioned into (indexed) strokes [96]. Compared to offline HWR, OnHWR has its own merits, e.g., the difficult segmentation of cursive written sequences into single characters. Many highly relevant handwriting problems in everyday life require both an informative representation of the writing and well-working classification algorithms [35]. Examples include the verification of signatures, the identification of writers, or the recognition of handwriting.

The representation of written text crucially depends on the way it has been recorded. Many recording systems make use of a stylus pen (a touch pen with a sensible magnetic mesh tip) together with a touch screen surface [2]. Systems for writing on paper are only prototypical, such as the ones used in [11,79,100] or the GyroPen [17] that provides a pen-like interaction from standard built-in sensors in modern smartphones. An advanced system for recording online HWR data was proposed by Ott et al. [65] who use a sensor-enhanced ballpoint pen that is extended with inertial measurement units (IMUs). The hand movement and velocity patterns with such a pen are different to air-writing [108]. In this paper, we propose a novel dataset collection of equations and words recorded with an IMU-enhanced pen. Using this pen allows an online representation of the accelerations, orientations and the pressure applied to the pen. Writing styles can thereby be characterized by an information-rich multivariate time-series (MTS). These datasets lay the foundation for HWR from pens with integrated sensors [17,45,62–65,79,100,104], a so far unsolved challenge in machine learning.

For machine learning tasks derived from online handwriting data, we distinguish between single-label prediction tasks (i.e., classifying characters, digits and symbols) and tasks to predict sequences of labels (i.e., words, sentences and equations). We here focus on the online seq2seq prediction task for writer-dependent (WD) and writer-independent (WI) classification, but also consider the single-label classification task. Seq2seq models in natural language processing (NLP) and speech recognition [86] are used to convert sequences of TYPE A to sequences of TYPE B (e.g., sentences from English to German). Many real-world datasets take the form of sequences, e.g., written texts, numbers, audio or video frame sequences. While many approaches build on language models or lexica [9,71,81,86] that outperform

model-free approaches for certain datasets (e.g., sentences), these approaches require additional efforts to properly deal with the data at hand. They cannot handle dialects and informal words off-the-shelf, do not recognize wrongly written words, and require a large corpus volume with large training times to achieve an acceptable accuracy [37]. Even with additional pre-processing, language models and lexica cannot (or only with high effort [92]) be applied to certain types of sequences, e.g., equations, as in our case. For our benchmark baselines we therefore resort to language- and lexicon-free approaches without token passing. More specifically, we provide an evaluation benchmark with CNNs combined with (bidirectional) LSTMs and TCNs, and an attention-based model for the seq2seq OnHWR, as well as several transformers for the single character-based classification task.

The remainder of the paper is organized as follows. We discuss related work in Sect. 2. Section 3 presents our novel collection of online handwriting datasets on sequence level. Section 4 introduces the suggested benchmark models; in particular, we propose several CNN architectures. In Sect. 5 we provide experimental results before we end with a conclusion in Sect. 6.

2 Background and related work

We will first provide an overview of available datasets of online handwriting datasets and explain the particularities for each one. Next, we discuss related methodological approaches to model such data. For a detailed overview of text classification methods we refer to [47,49].

2.1 Datasets

While there are many offline datasets, online data are rare [35]. To properly evaluate OnHWR methods, we need a multi-label online dataset that allows for the evaluation of tasks for both the writer-dependent and the writer-independent case. Table 1 gives an overview of available online datasets. For the single character prediction task, the Kuchibue [56,57], MRG-OHTC [53], CASIA [98] and OnHW-chars [65] datasets are available. While the OnHW-chars dataset is rather small, we provide single character-based datasets from a larger database. For our sequence-based method (i.e., a technique that predicts a whole sequence of characters), the IRONOFF [95], ICROW [78], IAM-OnDB [51], LMCA [42], ADAB [1], IBM-UB [84] and VNOnDB [58,59] word and sentence datasets can be used.

The commonly used IAM-OnDB [51] and VNOnDB [59] datasets only include online trajectory data written on a tablet. However, writing on even and smooth surfaces influences the writing style of the user [28]. To circumvent this disadvantage, we initially recorded a small character-only dataset

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

Table 1 Overview of state-of-the-art trajectory-based and our inertial-based online handwriting datasets

	Dataset	Content	Device	Writers	Statistics
Characters	Kuchibue [56,57]	Japanese characters	Tablet	120	10,154 × 120 char. patterns
	MRG-OHTC [53]	Tibetan characters	Tablet	130	910 character classes
	CASIA [98]	Chinese characters	Anoto pen on paper	1020	3.5 m characters
	OnHW-chars [65]	English characters	Sensor pen	119	31275 characters, 52 classes
	UNIPEN [33]	Sentences, words, characters	Pen-based computer	–	>12,000 chars. per writer
	CROHME [55]	Mathematical expressions	White-board, tablet	>100	9507 expressions
	IRONOFF [95]	French words, chars., digits	Trajectory, images	–	50,000 words, 32,000 chars.
	ICROW [78]	Dutch, Irish, Italian words	–	13,119 words	
	IAM-OnDB [51]	English sentences	Whiteboard	67	
	LMCA [42]	Arabic words, chars., digits	Tablet	197	82,272 words
Sequence	ADAB [1]	Arabic words	Tablet	55	30 k digits, 100 k chars., 500 w.
	IBM_UB_1 [84]	English words	Tablet	170	20,000+ words
	VNOnDB [58,59]	Vietnamese words, lines, paragr.	Notepad	43	6654 pages
	OnHW-equations	Equations written on paper	Tablet	200	110,746 words
	OnHW-words500	Repeated 500 words on paper	Sensor pen	55	10,720 equations, 15 classes
	OnHW-wordsRandom	Random words written on paper	Sensor pen	53	25,218 words, 59 classes
	OnHW-wordsTraj	Words written on a tablet	Sensor pen on tablet	54	14,645 words, 59 classes
	OnHW-symbols	Numbers, symbols on paper	Sensor pen	2	16,752 words, 52 classes
			Sensor pen	27	2326 characters, 15 classes

with a sensor-enhanced pen on usual paper in previous work [65]. In this paper we make use of this novel pen and record sequence-based samples for a comparison and evaluation benchmark with the trajectory-based IAM-OnDB (line level) and VNOnDB-words datasets. Hence, our datasets allow a broad research on sequence-based classification from sensor-enhanced pens and allow the connection between classical OnHW recognition on tablets with sensor-enhanced pens.

2.2 Methods

While hidden Markov models (HMMs) [3,8,18,19,22] have initially been applied to offline HWR, more recently, deep learning models became predominant, including convolutional neural networks (CNNs) [109], temporal convolutional networks (TCNs) [82,83], recurrent neural networks (RNNs) [20,31,68,70,85,105] including long short-term memorys (LSTMs), bidirectional LSTMs (BiLSTMs) [12,91] and multidimensional RNNs [32,97]. More recently, attention models further improved the classification accuracy of RNNs [10], but did not outperform previous approaches for OnHWR. Despite transformers [94] and its variants [13,36,38,44,90,101] got very popular for NLP [75] and image processing, these have so far only been applied to offline HWR [38]. The transformer by [66] is based on a language model and is used for Chinese text recognition. Similarly, variational autoencoders (VAEs), RNNs [29] and generative adversarial networks (GANs) [26] have been successfully applied for synthetic offline handwriting generation, but not for the online case so far. For the time-series classification task, standard convolutional architectures [25,34,72,103,113], spatio-temporal methods [6,15,21,39,40] and variants [24,87,89,99] as well as transformers [110] have been employed. In [65], we evaluated machine learning techniques, while in this paper we provide a broad evaluation benchmark on classical and novel time-series classification methods previously mentioned. While many approaches predict one class after the other, [14,54] predicted sequences similar to our approach. This is necessary to construct a suitable loss function described in the following. See Appendix 1 for a more detailed overview of related work.

Loss functions For sequence prediction the connectionist temporal classification (CTC) [30,31,43] loss combined with beam search [77] has extensively been used. The Edit distance (ED) [48] quantifies how dissimilar two strings are to one another by counting the minimum number of operations required to transform one string into the other. The ED allows deletion, insertion and substitution. However, the ED is a discrete function that is known to be hard to optimize. Ofitserov et al. [60] proposed a soft ED, which is a smooth approximation of ED that is differentiable. Seni et al. [80] used the

ED for HWR. We use the CTC loss for sequence prediction (see Sect. 4).

3 Datasets and evaluation methodology

Our datasets are a collection of existing and newly generated online handwriting recordings. Section 3.1 first describes our recording setup to create novel and information-rich datasets. Section 3.2 gives details about the properties of our different OnHW datasets and compares them to existing datasets. Section 3.3 proposes a set of evaluation metrics.

3.1 Recording setup

Our datasets are recorded with a sensor-enhanced pen developed by STABILO International GmbH that contains two accelerometers at the front and the back (3 axes each), one gyroscope (3 axes), one magnetometer (3 axes) and one force sensor at 100 Hz (see Fig. 2). The data recordings contain 14 measurements provided by the sensors: four sensor data signals (each in x, y and z direction), the force with which the pen tip touches the surface, and the timestep at which the tablet receives the data from the pen. Figure 1 shows an exemplary sensor signal from a written equation. Using the force sensor the sensor data allow to separate strokes well as the writer lifts the pen between every character (this is not possible for cursive writing, e.g., for words). In total, we let 131 adult writers participate in our data collection. For more information on the sensor pen and data acquisition, see Appendices 2 and 3.

3.2 Datasets

We propose a large set of four different sequence-based datasets (see the first four entries in Table 2): the **OnHW-equations** dataset was part of the *UbiComp 2021 challenge*¹ and is written by 55 writers and consists of 10 number classes and 5 operator classes (+, -, ., :, =). The dataset consists of a total of 10,713 samples. While in the **OnHW-words500** dataset only the same 500 words per each writer appear, in the **OnHW-wordsRandom** dataset every sample is randomly chosen from a large German and English word list. This allows the comparison of indirectly learning a lexicon of 500 words or, alternatively, completely lexicon-free learning. The OnHW-wordsRandom dataset (14,641 samples) is smaller than the OnHW-words500 dataset (25,218 samples), but contains longer words with a maximal length of 27 labels (19 labels for OnHW-words500). The train/validation split for the OnHW-words500 dataset is based on words for the

¹ UbiComp challenge: <https://stabilodigital.com/ubicomp-2021-challenge/>

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

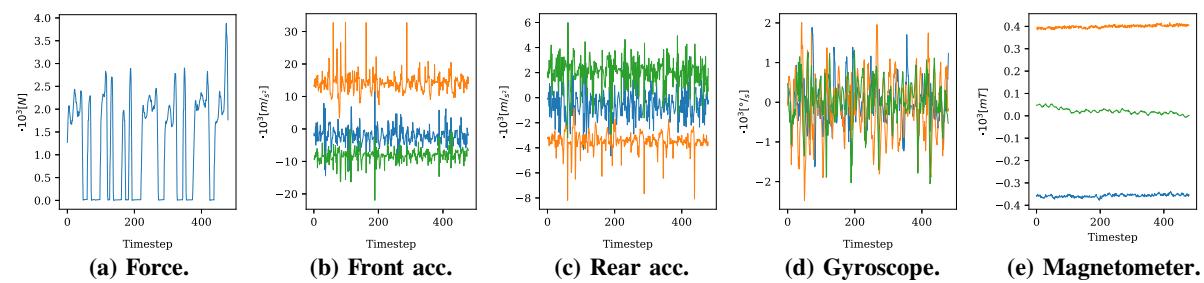


Fig. 1 Exemplary sensor data for the x-, y- and z-axis of the equation: 20583:70

Fig. 2 Sensor-enhanced pen

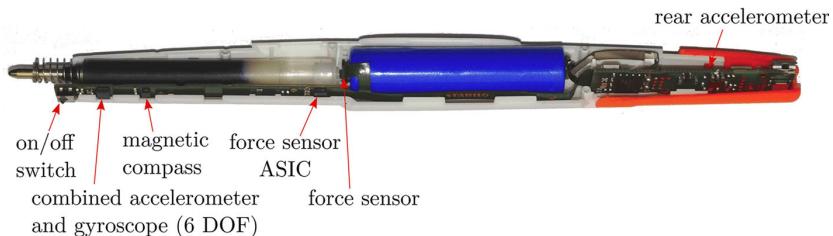


Table 2 Overview of our recordings from right-handed writers and state-of-the-art online handwriting datasets for writer-dependent (WD) and writer-independent (WI) tasks

Dataset	Number Writers	Number Classes	Maximal Length	Number samples		WI	Total Chars.
				Total	WD		
OnHW-equations	55	15	15	10,713	8595	2118	8610
OnHW-words500(R)	53	59	19	25,218	20,176	5042	19,918
OnHW-wordsRandom	54	59	27	14,641	11,744	2897	11,716
OnHW-wordsTraj	2	59	10	16,752	13,250	3502	—
OnHW-symbols	27	15	Single	2326	1853	473	1715
ICROW [78]	67	53	15	13,119	10,500	2619	10,524
IAM-OnDB [51]	197	81	64	10,773	8702	2071	8624
VNOnDB-words [59]	201	147	11	110,746	88,677	22,069	88,486
OnHW-chars [65]	119	52	Single	31,275	23,059	8216	23,059
						8216	31,275

Table 3 Overview of our datasets from left-handed writers for writer-dependent (WD) and writer-independent (WI) tasks

Dataset	Number Writers	Maximal Length	Number Samples			WI	Total Chars.
			Total	WD	WI		
OnHW-equations-L	4	15	843	677	166	543	300
OnHW-words500-L	2	19	1000	800	200	500	500
OnHW-wordsRandom-L	2	26	996	798	198	497	499
OnHW-symbols-L	4	Single	361	289	72	271	90
OnHW-chars-L [65]	9	Single	2270	1816	454	—	2270

For WD tasks a 80/20 train/validation split is used; for WI a dataset-specific split is used

Fig. 3 Exemplary online samples of our *OnHW-wordsTraj* dataset (left: tablet, middle: camera) and the IAM-OnDB [51] (line level) dataset (right)



WD task such that the same 400 words per writer are in the train set and the same 100 words per writer are in the validation set. For the WI task, the split is done by writer such that all 500 words of a single writer are either in the train or validation set. As it is more likely to overfit on the same words, the WD task of OnHW-words500 is more challenging compared to the OnHW-wordsRandom dataset. The **OnHW-words500R** dataset is a random split of OnHW-words500.

Additionally, we record the **OnHW-wordsTraj** dataset that consists of four different data sources. We replace the ink refill with a Wacom EMR module and record online trajectories at 30Hz on a Samsung Galaxy Tab S4 tablet along with the sensor data. Four cameras pointed on the pen to record the movement of the pen tip at 60Hz. We manually label the pixels of 100 random images of the recorded videos in the classes “pen”, “pen tip” and “background” and train U-Net [76] to predict the pen tip pixels from all images. From this we derive the pen tip trajectory in camera coordinates. Two persons wrote 4257 words in total that results in 16,752 camera samples. With this dataset it is possible to compare results from traditional online trajectory datasets (written on a tablet) with our online sensor pen datasets. Figure 3 exemplarily compares the trajectory and camera data of the OnHW-wordsTraj dataset with the IAM-OnDB [51] dataset. Table 3 gives a dataset overview of left-handed writers. Sample sizes are smaller and ranges between around 3% and 13.4% of the sample sizes of right-handed datasets. For our benchmark, we consider right- and left-handed writers separately and will publish right- as well as left-handed datasets for future research.²

Figure 4 compares statistical properties, i.e., the number of samples, sample lengths and character distributions, between our dataset and the state-of-the-art datasets. The IAM-OnDB (line level) and VNOnDB-words datasets consist of more samples and total number of characters compared to our OnHW datasets, but at the same time use a higher number of classes (81 and 147). The IAM-OnDB samples have higher lengths (up to 64), and the VNOnDB samples have smaller lengths (up to 11) (see Fig. 4a). The VNOnDB dataset is equally distributed compared to other words datasets (see Fig. 4c), while numbers appear more often than operators in our OnHW-equations dataset (see Fig. 4b). See Appendices A.4 and A.5 for more details on our datasets.

Datasets for single character classification For the OnHW-equations dataset, it is possible to split the sensor sequence based on the force sensor as the pen is lifted between every single character. This approach provides another useful dataset for a single character classification task. We set split constraints for long tip lifts and recursively split these

² www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html

sequences by assigning a possible number of strokes per character. This results in a total of 39,643 single characters. Furthermore, we recorded the **OnHW-symbols** dataset with the same labels (numbers 0 to 9 and operators +, -, ·, :, =), written by 27 writers and a total of 2326 single characters. Figure 5 compares the distribution of sample numbers for the OnHW-chars [65] (characters) and OnHW-symbols as well as split OnHW-equations (numbers, symbols) datasets. While the samples are equally distributed for small and capital characters (≈ 600 per character), the numbers and symbols are unevenly distributed for the split OnHW-equations dataset (similar to Fig. 4b).

3.3 Evaluation metrics

We define a set of task-specific seq2seq and single character-based evaluation metrics that are commonly used in the community. Metrics for seq2seq evaluation are the **character error rate** (CER) and **word error rate** (WER) that are based on the **Edit distance** (ED). The ED is the minimum number of substitutions S , insertions I and deletions D required to change the sequences $\mathbf{f} = (f_1, \dots, f_r)$ into $\mathbf{g} = (g_1, \dots, g_n)$ with lengths r and n , respectively. The ED is defined by

$$\text{ED}_{i,j} = \begin{cases} \text{ED}_{i-1,j-1} & \text{for } f_i = g_j \\ \min \begin{cases} \text{ED}_{i-1,j} + D(f_i) \\ \text{ED}_{i,j-1} + I(g_j) \\ \text{ED}_{i-1,j-1} + S(f_i, g_j) \end{cases} & \text{for } f_i \neq g_j \end{cases} \quad (1)$$

for $1 \leq i \leq r, 1 \leq j \leq n$, $\text{ED}_{i,0} = \sum_{k=1}^i D(f_k)$ for $1 \leq i \leq r$, and $\text{ED}_{0,j} = \sum_{k=1}^j I(g_k)$ for $1 \leq j \leq n$ [16]. We define the $\text{CER} = \frac{S_c + I_c + D_c}{N_c}$ as the ED, the sum of character substitutions S_c , insertions I_c and deletions D_c , divided by the total number of characters in the set N_c . Similarly, the $\text{WER} = \frac{S_w + I_w + D_w}{N_w}$ is computed with word operations S_w , I_w and D_w and number of words in the set N_w [38]. For single character evaluation, we use the **character recognition rate** (CRR) that is the number of correctly classified characters divided by the total number of characters in the test set.

4 Benchmark methods

This section formally defines the seq2seq classification task and our loss functions. We propose our architecture for HWR from IMU-enhanced pens and describe our data augmentation techniques.

Sequence-based classification task An MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $l \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l}), i \in \{1, \dots, m\}$, where m is the length

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

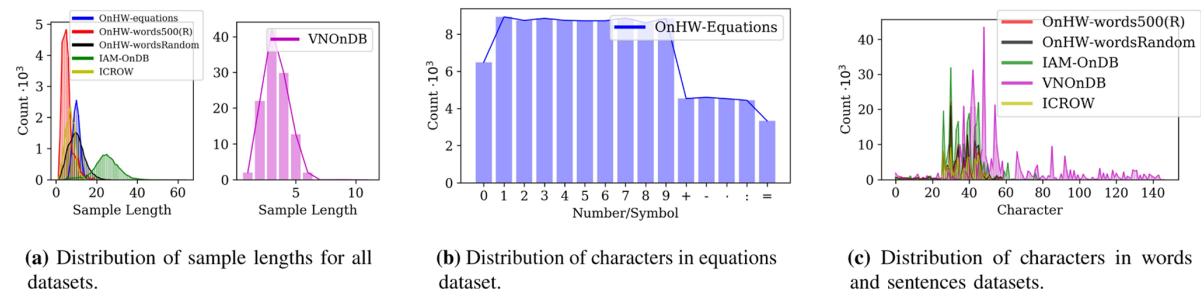


Fig. 4 Properties of our and state-of-the-art datasets

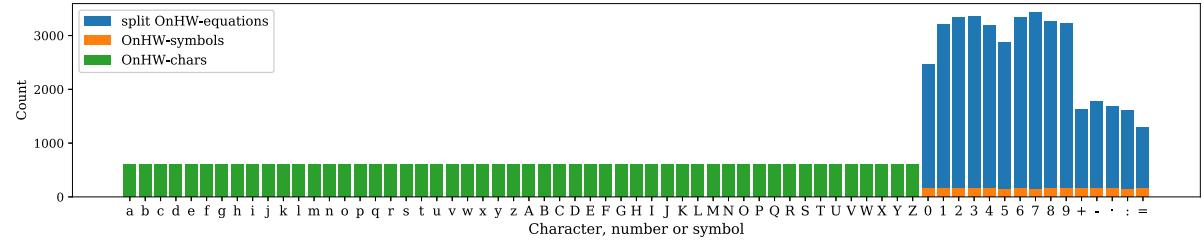


Fig. 5 Distribution of samples for the OnHW-chars, OnHW-symbols and split OnHW-equations datasets

of the time-series that is variable and l is the number of dimensions. Each MTS is associated with \mathbf{v} , a sequence of L class labels from a pre-defined label set Ω with K classes. For our classification task, $\mathbf{v} \in \Omega^L$ describes words and equations. The training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_n\} \in \mathbb{R}^{n \times m \times l}$, where n is the number of time-series, and the corresponding labels $\mathcal{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\} \in \Omega^{n \times L}$. The aim of the MTS classification task is to predict an unknown class label for a given MTS. We train the classifier using the loss $\mathcal{L}_{CTC}(\mathcal{U}, \mathcal{V})$ [30].

Character-based classification task In contrast to the sequence-based classification task, corresponding labels \mathcal{V} for the character-based classification task are of length $L = 1$. We define $p(i|\mathbf{u})$ to be the predicted probability for the i th class and $q(i|\mathbf{u})$ to be the true class distribution. We train the classifier using the cross-entropy loss and variants against overconfidence and class imbalance [50, 67, 73, 88, 102, 112].

Sequence-based loss The CTC loss is a solution to avoid pre-segmentation of the training samples. The key idea of CTC is to transform the network outputs into a conditional probability distribution over label sequences. An intermediate label representation allows repetitions of labels and occurrences of blank labels to identify no output label. Hence, the network with the CTC loss has a softmax output layer with one more unit than there are labels. These outputs define the probabilities of all possible ways to align all label sequences with the input sequence. [30]

Character-based losses We use the categorical cross-entropy (CCE) loss defined by

$$\mathcal{L}_{CCE}(\mathcal{U}, \mathcal{V}) = -\frac{1}{K} \sum_{i=1}^K q(i|\mathbf{u}) \log p(i|\mathbf{u}) \quad (2)$$

for model training. Samples with softmax outputs that are less congruent with provided labels are implicitly weighted more than confident sample predictions (more emphasis is put on difficult samples with CCE). Hence, more emphasis is put on difficult samples, which can cause overfitting to noisy labels [102, 112]. To account for this imbalance, we modify the CCE loss such that it down-weights the loss assigned to well-classified examples. We use the Focal loss (FL) [50] defined by

$$\begin{aligned} \mathcal{L}_{FL}(\mathcal{U}, \mathcal{V}, \alpha, \gamma) \\ = -\frac{1}{K} \sum_{i=1}^K \alpha_i (1 - p(i|\mathbf{u}))^\gamma q(i|\mathbf{u}) \log p(i|\mathbf{u}), \end{aligned} \quad (3)$$

with class balance factor $\alpha \in [0, 1]$, and the modulating factor $(1 - p(i|\mathbf{u}))^\gamma$ with focusing parameter $\gamma \geq 0$. As alternative, we apply label smoothing (LSR) [67] that prevents overconfidence by applying a confidence penalty through a regularization term, yielding

$$\mathcal{L}_{LSR}(\mathcal{U}, \mathcal{V}, \beta) = -\frac{1}{K} \sum_{i=1}^K \log p(i|\mathbf{u}) - \beta H(p(i|\mathbf{u}))$$

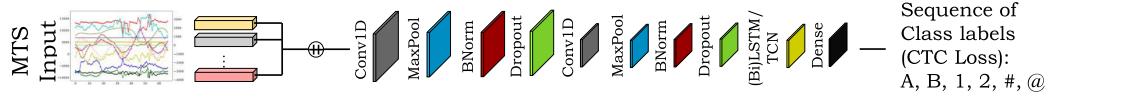


Fig. 6 Overview of our CNN architecture in combination with a (Bi)LSTM or a TCN

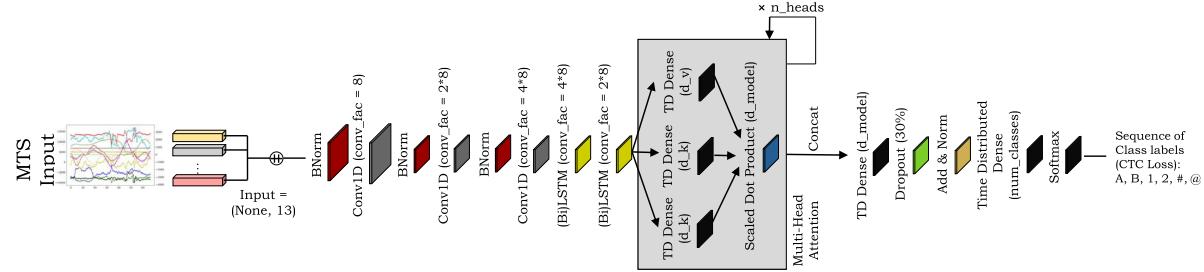


Fig. 7 Overview of our architecture with multi-head attention

$$= -\frac{1}{K} \sum_{i=1}^K \log p(i|\mathbf{u}) - D_{KL}(x||p(i|\mathbf{u})), \quad (4)$$

with β the strength control of the confidence penalty. Label smoothing is equivalent to an additional Kullback-Leibler (KL) divergence term between a uniformly distributed random variable x and the network's predicted distribution p . The bootstrapping approach [73] is another alternative for each mini-batch. The soft bootstrapping loss (SBS) is

$$\mathcal{L}_{\text{SBS}}(\mathcal{U}, \mathcal{V}, \beta) = -\frac{1}{K} \sum_{i=1}^K [\beta q(i|\mathbf{u}) + (1 - \beta)p(i|\mathbf{u})] \log p(i|\mathbf{u}), \quad (5)$$

for predicted class probabilities p with weighting parameter β , while the hard bootstrapping loss (HBS)

$$\mathcal{L}_{\text{HBS}}(\mathcal{U}, \mathcal{V}, \beta) = -\frac{1}{K} \sum_{i=1}^K [\beta q(i|\mathbf{u}) + (1 - \beta)z_i] \log p(i|\mathbf{u}) \quad (6)$$

uses the maximum a posteriori (MAP) estimation of p given \mathbf{u} , with $z_i := \mathbb{1}[i = \arg \max q_l, l = 1, \dots, K]$. MAP treats every sample equally for a higher robustness against noisy labels. This can lead to longer training times to reach convergence and can make optimization more difficult [112]. The generalized cross-entropy (GCE) [112] loss

$$\mathcal{L}_{\text{GCE}}(\mathcal{U}, \mathcal{V}, \alpha) = -\frac{1}{K} \sum_{i=1}^K \frac{1 - p(i|\mathbf{u})^\alpha}{\alpha} \quad (7)$$

with $\alpha \in (0, 1]$ uses a negative Box-Cox transformation to combine benefits of the MAP and the CCE. The symmetric cross-entropy (SCE) [102] is

$$\mathcal{L}_{\text{SCE}}(\mathcal{U}, \mathcal{V}, \alpha, \beta) = \alpha \mathcal{L}_{\text{CCE}}(\mathcal{U}, \mathcal{V}) + \beta \mathcal{L}_{\text{RCE}}(\mathcal{U}, \mathcal{V}) \quad (8)$$

based on the reverse cross-entropy (RCE) loss

$$\mathcal{L}_{\text{RCE}}(\mathcal{U}, \mathcal{V}) = -\frac{1}{K} \sum_{i=1}^K p(i|\mathbf{u}) \log q(i|\mathbf{u}), \quad (9)$$

aims for a more effective and robust learning, where α mitigates the overfitting of CCE and β allows for flexible exploration of the RCE. Furthermore, we make use of the joint optimization (JO) [88], which overcomes the noisy labels problem by learning network parameters and labels jointly. The loss is defined by

$$\mathcal{L}_{\text{JO}}(\Theta, \mathcal{V}|\mathcal{U}, \alpha, \beta) = \mathcal{L}_{\text{CCE}}(\Theta, \mathcal{V}|\mathcal{U}) + \alpha \mathcal{L}_p(\Theta|\mathcal{U}) + \beta \mathcal{L}_e(\Theta|\mathcal{U}) \quad (10)$$

with regularization losses \mathcal{L}_p and \mathcal{L}_e , and network parameters Θ .

Architectures We propose two different architectures for seq2seq sensor signal classification. For the first method (see Fig. 6), a convolution block consisting of 1D convolutions (200 filter, kernel size 4), max pooling (pool size 2), batch normalization and dropout (with rate 0.2) layers is used. One TCN layer of 100 units, one LSTM layer of 100 units or two BiLSTM layers, each with 60 units, follow to extract the temporal context [74]. While we use \tanh activations for BiLSTM layers, we choose ReLU for the TCN and LSTM layers. A dense layer with 100 units with the CTC loss predicts a sequence of class

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

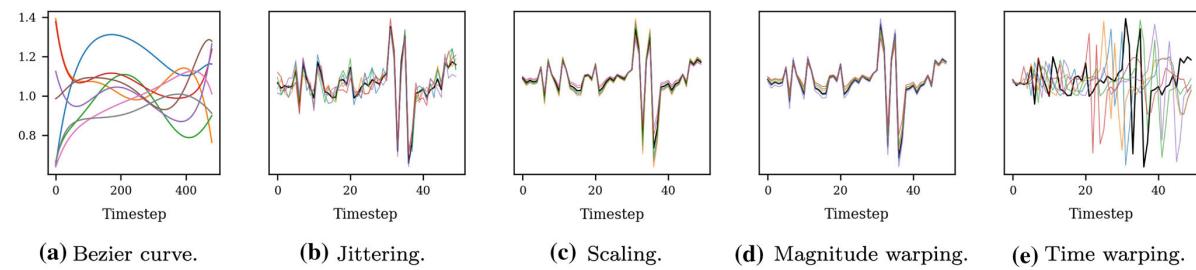


Fig. 8 Data augmentation methods of the original sensor sample (black)

labels. Second, we implement an attention-based network (see Fig. 7) that consists of an encoder with batch normalization, 1D convolutional and (Bi)LSTM layers. These map the input sequence $\mathbf{U} \in \mathbb{R}^{m \times l}$ to a sequence of continuous representations \mathbf{z} . A transformer transforms \mathbf{z} using n_{head} stacked multi-head self-attention $\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$ with $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$. The attention consists of point-wise, fully connected time-distributed layers followed by a scaled dot product layer and layer normalization [5] with d_{model} output dimension [94]. $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$, where $W_i^Q, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, and $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$. The attention can be described as mapping a set of key-value pairs of dimension d_v and a query of dimension d_k to an output, and is computed by $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$. The matrices Q , K and V are a set of queries, keys and values.

Data augmentation As the size of the datasets is limited, data augmentation is a critical pre-processing step for networks to prevent overfitting and improve generalization. However, it is not obvious how to carry out label-preserving augmentation in some domains, i.e., scaling of acceleration signals [93]. We apply the following different data augmentation methods for wearable sensor data on each sensor channel at 50% probability. **Time warping** perturbs the temporal location by smoothly distorting the time intervals between samples that, e.g., simulates different sampling rates through time shifts of the connection between device and tablet. **Scaling** changes the magnitude of the data in a window by multiplying by a random scalar $\sigma = \pm 0.1$ that augments drifts in the signals. **Shifting** adds a random value $\alpha = \pm 200$ to the force data and $\alpha = \pm 20$ to the other sensor data. While **jittering** is a way of simulating additive sensor noise by adding a multiple $\sigma = \pm 0.1$ of the standard deviation to all sensor channels, **magnitude warping** changes the magnitude by convolving the data window with a smooth curve varying around [0.7, 1.3] (only for the accelerometer data). For time and magnitude warping, the data are augmented by Bézier curves in the interval $[1 - \sigma, 1 + \sigma]$ that are generated based on 10 random points. As one sample is represented by

a sequence of characters and a sample cannot be split into sub-sequences, applying cropping and permutation augmentation is not possible. Figure 8 zooms into the augmented sensor data of the x-axis signal from Fig. 1.

5 Experiments

This section provides evaluation results for the seq2seq (Sect. 5.1) and the single character-based classification task (Sect. 5.2), and evaluates left-handed datasets (Sect. 5.3). We propose a writer-dependent evaluation in Sect. 5.4.

Hardware and training setup For all experiments we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the Adam optimizer with a learning rate of 10^{-4} . We run each experiment for 1000 epochs with a batch size of 50 (unless stated differently) and report results for the best epoch. We split each dataset into five approx. 80/20 train/validation splits and report the mean and standard deviation of the WER and CER. We use our OnHW-equations, OnHW-words500(R), OnHW-wordsRandom and OnHW-wordsTraj as well as the IAM-OnDB [51] and VONONDB-words [59] datasets for the sequence-based classification task, and the OnHW-symbols, split OnHW-equations and OnHW-chars [65] datasets for the single character-based classification task. Each model is trained from scratch for every dataset. We make use of the time-series classification toolbox tsai [61] that contains a variety of state-of-the-art techniques [6, 15, 21, 24, 25, 34, 39, 72, 87, 89, 99, 103, 110, 113].

5.1 Seq2seq task evaluation

Method and architecture evaluation We first evaluate our CNN and attention-based models for the seq2seq classification task. A summary of results is given in Table 4. For all datasets our CNN+BiLSTM model significantly outperforms the CNN+LSTM and CNN+TCN models. The attention-based model performs poorly on large datasets (OnHW-equations, words500(R), wordsRandom), but yields better

Table 4 Evaluation results (WER, CER) in % (mean and standard deviation) for our OnHW-equations, OnHW-words500(R), OnHW-wordsRandom and OnHW-wordsTraj writer-dependent (WD) and writer-independent (WI) datasets, and the publicly available IAM-OnDB [51] (line level) and VNONDB-words [59] datasets

Method	Met- ric	OnHW-equations			OnHW-words500(R)			OnHW-wordsRandom					
		WD	WER	CER	WD	WER	CER	Random	WER	CER	WD	WER	CER
CNN+LSTM	Mean	22.96	3.50	69.22	18.11	80.70	28.41	93.30	48.24	76.80	23.73	82.29	17.90
CNN+LSTM	STD	1.83	0.38	7.91	5.20	3.32	2.50	1.13	4.59	0.34	0.23	8.49	1.66
CNN+BiLSTM	Mean	13.19	1.78	55.25	12.98	51.95	17.16	60.91	27.80	18.77	5.20	41.27	7.87
CNN+TCN	Mean	28.57	4.29	82.06	23.95	63.51	21.07	90.54	49.53	62.61	19.13	83.16	19.26
CNN+TCN	STD	1.16	0.23	6.14	4.44	11.81	4.37	5.56	7.93	12.03	3.90	7.82	2.42
Attention-based model	Mean	73.69	16.45	87.48	27.45	88.34	45.70	83.53	42.42	78.53	35.05	96.33	42.14
Attention-based model	STD	2.55	1.04	2.19	2.33	1.74	1.46	2.42	2.42	5.21	2.12	1.96	1.73
InceptionTime [25]	Mean	20.72	2.92	60.24	14.71	41.92	12.08	76.84	35.07	40.18	11.39	63.04	12.81
InceptionTime [25]	STD	0.58	0.18	8.81	4.75	2.38	0.64	2.92	5.63	0.62	0.24	0.99	0.21
InceptionTime [25]	Mean	19.48	2.72	60.90	14.29	53.34	16.24	78.22	36.85	47.52	13.87	65.68	13.63
InceptionTime [25]	STD	0.29	0.13	7.87	4.61	4.34	0.71	3.53	6.53	2.02	0.83	1.31	0.36
(32, 6)	Mean	12.94	1.77	52.40	12.23	37.12	12.96	62.09	26.36	21.34	5.34	42.88	7.19
(32, 6)	STD	0.33	0.12	8.09	4.71	2.11	0.55	5.66	2.21	0.56	0.20	1.27	0.25
(96, 11)	Mean	12.06	1.65	49.92	11.28	43.22	13.07	61.62	26.08	21.18	5.35	39.14	6.39
(96, 11)	STD	0.32	0.10	7.78	4.20	2.93	0.79	5.39	6.27	0.84	0.26	0.83	0.13
XceptionTime [72]	Mean	38.66	5.67	71.06	17.52	49.10	15.07	78.54	36.80	45.84	13.81	69.20	15.60
XceptionTime [72]	STD	0.80	0.20	5.70	4.56	2.79	0.57	3.55	6.14	0.48	0.14	0.55	0.21
(144)	Mean	38.40	5.71	70.56	17.47	51.62	16.24	80.00	38.06	46.44	14.26	71.74	16.77
(144)	STD	1.14	0.21	5.07	4.32	4.00	1.37	2.96	5.55	0.45	0.11	0.72	0.31
ResNet [103]	Mean	39.36	5.78	87.10	27.56	90.30	44.23	95.90	58.61	77.02	27.64	92.50	27.37
ResNet [103]	STD	2.44	0.61	4.77	4.10	7.29	13.13	0.95	3.35	4.28	3.38	0.53	0.38
ResNet [103]	Mean	37.50	5.50	84.02	25.84	79.54	28.19	96.66	59.76	79.04	28.16	91.36	25.31
ResNet [103]	STD	3.10	0.59	9.33	5.97	3.11	1.51	0.34	2.58	0.48	0.37	0.84	0.97
(144) +BiLSTM	Mean	81.92	18.20	98.50	45.59	94.42	48.01	98.92	70.68	92.32	44.81	98.68	41.78
(144) +BiLSTM	STD	1.29	0.79	0.87	4.60	1.57	2.04	0.16	2.08	1.89	0.17	0.74	0.40

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

Method	Met- ric	OnHW-equations			OnHW-words500(R)			OnHW-wordsRandom					
		WD	WER	CER	WI	WER	CER	Random	WER	CER	WI	WER	CER
ResCNN [113] (144)+BiLSTM	Mean	87.66	23.24	99.54	51.77	94.56	48.59	98.86	70.06	93.80	45.59	99.10	43.33
	STD	2.33	1.77	0.43	4.67	1.41	1.77	0.33	1.57	0.49	1.84	0.24	0.43
FCN [103]	Mean	91.62	24.66	99.46	53.84	96.82	54.89	99.34	75.46	96.74	54.58	99.54	48.54
	STD	0.92	1.04	0.37	2.73	0.66	0.88	0.14	2.80	0.14	0.55	0.08	0.70
LSTM-FCN [39]	Mean	90.82	24.47	99.44	52.49	96.18	52.53	99.48	76.94	95.82	51.50	99.48	50.06
	STD	1.40	1.44	0.40	3.96	1.06	1.52	0.07	1.80	0.51	1.33	0.07	0.64
GRU-FCN [21]	Mean	89.12	23.03	99.32	52.01	96.78	55.11	99.46	76.05	96.66	54.32	99.60	51.97
	STD	1.53	1.08	0.53	3.93	0.93	1.55	0.10	1.40	0.51	1.43	0.11	1.18
MLSTM-FCN [40]	Mean	87.18	21.75	99.28	48.82	98.46	70.02	99.30	77.03	97.66	63.19	99.36	47.88
	STD	1.67	0.96	0.35	3.68	2.08	10.30	0.11	1.70	1.89	10.21	0.05	0.93
MGRU-FCN [40]	Mean	88.64	22.50	99.34	50.56	96.80	55.22	99.4	74.34	96.16	53.02	99.38	49.32
	STD	0.99	0.90	0.60	4.43	0.89	2.05	0.11	2.21	0.64	1.26	0.13	1.14
Method	Met- ric	OnHW-wordsTraj ¹			IMU-OnDB [51]			VNOnDB-words [59]					
		IMU Camera ²	WER	CER	IMU WER	WER	CER	Trajectory	WD	WER	WI	WER	CER
CNN+LSTM	Mean	60.50	14.93	57.00	8.95	61.10	10.66	83.14	11.23	84.56	12.96	60.54	26.12
	STD	—	3.03	0.95	8.74	2.50	1.30	0.65	1.66	1.35	13.77	7.57	8.62
CNN+BiLSTM	Mean	26.22	8.54	16.52	2.79	11.77	2.07	65.91	6.94	72.42	9.11	15.54	6.71
	STD	—	2.38	0.61	2.23	0.64	1.08	0.27	2.75	1.22	0.67	0.25	1.24
CNN+TCN	Mean	64.00	16.10	67.47	11.40	69.40	23.94	87.07	12.97	87.18	14.32	41.70	16.98
	STD	—	—	7.12	4.80	16.61	27.28	3.00	1.49	2.91	1.74	8.43	3.57
Attention-based model	Mean	60.99	17.21	74.80	16.74	33.50	5.78	—	—	—	—	—	—
	STD	—	2.09	0.74	4.45	1.01	—	—	—	—	—	—	—
InceptionTime [25] (96, 11)	Mean	59.30	51.91	34.64	2.70	12.32	2.14	73.50	8.72	78.36	10.99	19.84	7.79
	STD	—	—	1.74	0.47	1.86	0.56	2.71	0.82	3.35	1.47	1.61	0.61
InceptionTime [25] (96, 11)+BiLSTM	Mean	99.75	75.76	16.35	2.56	11.34	2.00	71.46	8.23	75.14	9.91	23.02	9.35
	STD	—	—	2.23	0.53	1.55	0.47	1.87	0.42	2.2	1.04	5.25	2.12

¹Best results are in bold²Averaged results over two writers.²Only one train/validation split

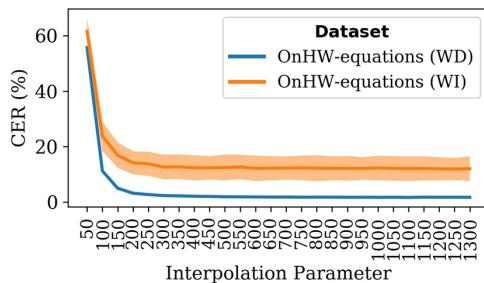


Fig. 9 CER of InceptionTime [25] for different interpolated time-series lengths on OnHW-equations dataset

results than the CNN+TCN on our OnHW-wordsTraj camera-based dataset and outperforms the CNN+LSTM and CNN+TCN models on the trajectory-based dataset. The CNN+BiLSTM model achieves a very good CER of 1.78% on the OnHW-equations WD dataset that increases to 12.98% for the WI task. For the words, IAM-OnDB and VONonDB datasets, the WI classification task is more difficult. While we achieve very low CERs, the WERs are higher as no lexicon or language model is used. While for the OnHW-wordsRandom dataset the CER of 7.87% for the WD task increases notably to 35.22% for the WI task, the difference for the OnHW-words500 dataset is smaller (17.16% CER for the WD task and 27.80% for the WI task) as words in the validation set do not appear in the training set (WD task). For the OnHW-words500R dataset, the CER decreases to 5.20% as the split is randomly shuffled. Our OnHW-wordsTraj dataset allows a comparison of three recording devices (i.e., trajectory, IMU and camera). From the CNN+BiLSTM model we see that the spatio-temporal trajectory-based classification task is easier than OnHWR from IMU-enhanced pens. Furthermore, it is challenging to learn the transformation from camera to paper plane.

Comparison to state-of-the-art techniques For comparison, we train nine different well-established time-series classification architectures on our OnHW datasets and InceptionTime [25] on the tablet datasets. For these methods we interpolate and zero pad the time-series to 800 timesteps to obtain a fixed sequence length. We use linear spline interpolation. In total, 800 timesteps lead to a low CER (see Fig. 9), while above 800 timesteps the training time significantly increases. As these methods are introduced for classifying single labels (not sequences of labels), we replace the last linear layer with a max pooling layer (of kernel size 4), a dropout layer (40%) and an 1D convolutional layer (kernel size 1 and channels are the number of class labels). Similar to our approaches, we further add two BiLSTM layers each of size 60. InceptionTime is an ensemble of CNNs inspired by Inception-v4. As its default parameters (nf of 32 and $depth$

of 6) lead to inferior performance compared to our methods, we perform a large hyperparameter search for $depth$ (between 3 and 12) and nf (16, 32, 64, 96 and 128) with and without BiLSTMs for the WD and WI tasks (see Fig. 10). On the WD dataset, a higher nf and greater $depth$ leads to a lower CER. For the WI task, the model tends to overfit on specific writers for larger models, and hence, the error rates are constant for nf between 64 and 128, while the CER still decreases for a greater $depth$. For nf of 32 and $depth$ of 11, InceptionTime+BiLSTM can marginally outperform our CNN+BiLSTM model on the OnHW-equations dataset (1.65% CER WD and 11.28% CER WI) and is notably better on the OnHW-words500 (WD) dataset (12.96% CER) without the two additional BiLSTM layers, but is on par with our CNN+BiLSTM model on the WI task (26.08% CER) and yields marginally higher error rates on the random splits. Results further suggest that the performance strongly depends on the network size. XceptionTime [72] consists of depthwise separable convolutions and adaptive average pooling to capture both temporal and spatial contents. We search for the hyperparameter hf (see Fig. 11) and set $nf = 144$. The small FCN [103] model yields high error rates, but ResNet [103] (based on FCN) enables the exploitation of class activation maps to find contributing regions in the raw data and improves FCN. ResCNN [113] integrates residual networks with CNNs. We set also $nf = 144$ for ResCNN and ResNet (see Fig. 12), which perform similar, but cannot outperform XceptionTime on our datasets. While additional BiLSTM layers improve the results of InceptionTime, the error rates for XceptionTime, ResNet and ResCNN decrease with additional BiLSTM layers. The univariate models LSTM-FCN [39] and GRU-FCN [21] as well as the multivariate models MLSTM-FCN [40] and MGRU-FCN [40] that augment the fully convolutional block with a squeeze-and-excitation block improve the FCN results, but are not complex enough to outperform other architectures on our datasets. In general, word beam search [77] did not improve results and even leads to degraded performance. See Appendix 7 for more evaluation details and a comparison to state-of-the-art techniques.

Influence of data augmentation We train the CNN+LSTM model on the OnHW-equations dataset with the augmentation techniques described in Sect. 4. Results are given in Table 5. The baseline WER of 22.96% (WD) can be improved with all augmentation techniques, while the WI error of 69.21% is only affected by *time warping* and *jittering*. The most notable improvement is given by *time warping* with 20.90% for the WD task and 64.10% for the WI task. *Interpolation* to 1,000 timesteps did not improve the accuracy, and *normalization* to $[-1, 1]$ deteriorates training performance. Figure 13 shows augmentation results and combinations of these for InceptionTime on the OnHW-

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

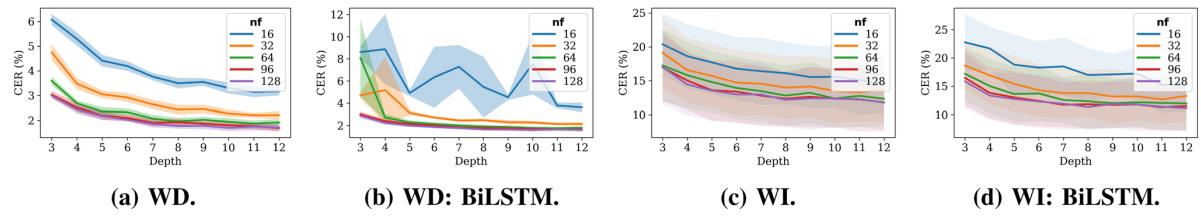


Fig. 10 Hyperparameter search of *depth* and *nf* for InceptionTime [25] with and without BiLSTM on the OnHW-equations datasets averaged over 5 splits

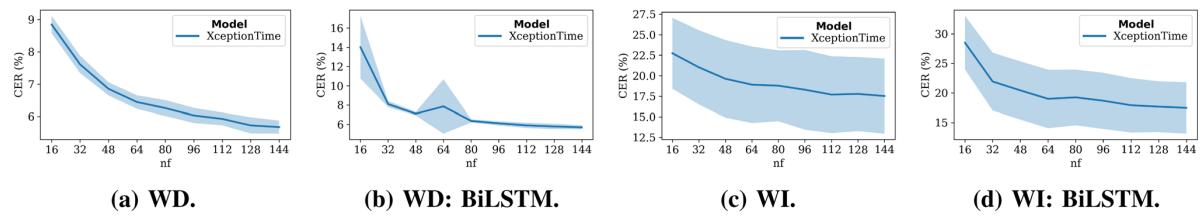


Fig. 11 Hyperparameter search of *nf* for XceptionTime [72] with and without BiLSTM on the OnHW-equations datasets averaged over 5 splits

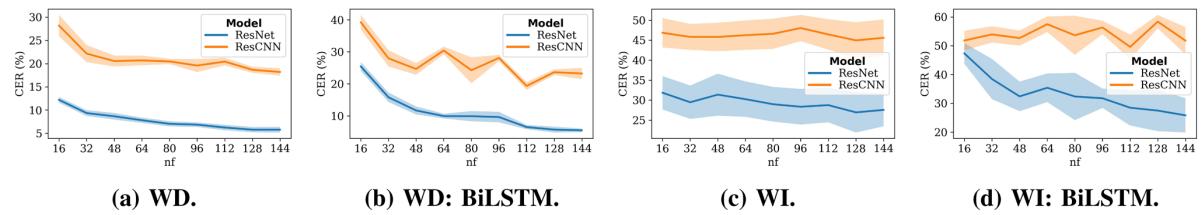


Fig. 12 Hyperparameter search of *nf* for ResNet [103] and ResCNN [113] with and without BiLSTM on the OnHW-equations datasets averaged over 5 splits

Table 5 Evaluation results (WER, CER) in % (mean and standard deviation over fivefold splits) for different augmentation techniques and sensor choices for the OnHW-equations dataset

Augmentation Technique	Sensors	WD				WI			
		WER Mean	WER STD	CER Mean	CER STD	WER Mean	WER STD	CER Mean	CER STD
None	All	<u>22.96</u>	1.83	<u>3.50</u>	0.38	<u>69.21</u>	7.91	<u>18.11</u>	5.20
Scaling (S)	All	22.70	0.40	3.43	0.22	69.70	7.90	18.80	5.84
Time Warping (TW)	All	20.90	0.83	3.18	0.27	64.10	5.51	15.26	2.27
Jittering (J)	All	22.87	0.75	3.47	0.33	68.14	10.03	18.68	7.18
Magnitude Warping (MW)	All	22.88	1.21	3.53	0.29	76.80	8.35	18.47	5.21
Shifting (SH)	All	22.40	1.12	3.43	0.24	69.81	7.59	18.80	4.88
Interpolation	All	25.04	0.92	3.96	0.32	70.50	8.30	19.42	5.96
Normalization	All	55.26	2.04	7.97	0.51	82.48	8.74	22.71	5.04
None	w/o Magnetometer	22.60	1.51	3.44	0.36	63.48	8.32	16.07	4.73
None	w/o Front Accelerometer	21.36	0.60	3.28	0.29	70.24	8.25	19.55	5.52
None	w/o Rear Accelerometer	23.20	0.86	3.57	0.26	68.30	8.14	16.64	5.40
None	w/o Mag., w/o Front Acc.	22.46	1.55	3.41	0.38	69.12	8.40	17.31	4.02

Bold are baseline improvements

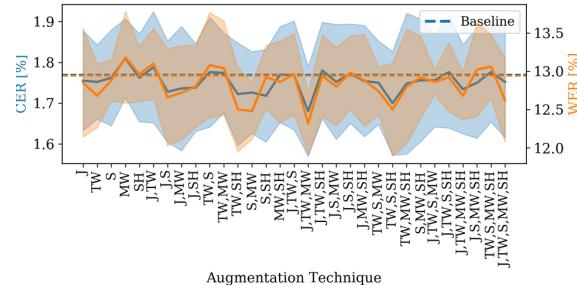


Fig. 13 Augmentation results for InceptionTime on the OnHW-equations (WD) dataset over five splits

equations WD dataset. Here, the baseline CER of 1.77% and WER of 12.94% can be notably improved by *time warping* as a single augmentation (comparable to our CNN+LSTM). The combination of *jittering* and *time and magnitude warping* yields the highest error rate reduction.

Influence of sensor dropping We train the OnHW-equations dataset and drop data from single sensors, e.g., the front or rear accelerometer or the magnetometer data, in order to evaluate the influence of each sensor, see Table 5. Only dropping the front accelerometer (WD) and the rear accelerometer (WI) decreases the WER and CER, which could also be attributed to the smaller dataset size while leaving the architecture unchanged. Without magnetometer the WER improves for the WI task as the magnetic field changes with the recording location, but keeps constant for the same writer. Dropping the force sensor leads to a significant higher classification error as the force sensor provides information that allows a segmentation of strokes.

5.2 Single character task evaluation

Method and architecture evaluation We use our OnHW-symbols and split OnHW-equations datasets, the combination of both (samples randomly shuffled) and the OnHW-chars [65] dataset, and interpolate the single characters to the longest single character of the dataset (64 for characters and 79 for number/symbols). We train our network proposed in Fig. 6 with one additional dense layer of 100 units. For all methods we use the categorical CE loss for training and the CRR for evaluation. Network parameter choices are described in Appendix 6. The results are summarized in Table 6. We also compare to state-of-the-art results provided in [65]. While GRU [15] yields very low accuracies for all datasets, standard LSTM units (2 and 3 stacked layers), BiLSTM units and TCNs can increase the CRR. Further, FCN [40] and the spatio-temporal variants RNN-FCN [40], LSTM-FCN [39] and GRU-FCN [21] as well as the mul-

tivariate variants MRNN-FCN, MLSTM-FCN [40] and MGRU-FCN [40] yield better results. MLSTM-FCN [40] with a standard or attention-based LSTM and with/out a squeeze-and-excitation (SE) block achieves high accuracies, but cannot improve over state-of-the-art results achieved by [65]. Due to minor and inconsistent changes in performance, it is not possible to make a statement about the importance of the SE block and the attention-based LSTM. The networks based on CNNs, i.e., ResCNN [113], ResNet [103], XResNet [34], InceptionTime [25] and XceptionTime [72], can partly outperform the FCN variants. For XResNet, a smaller depth of the network is preferable, while for InceptionTime the greater *depth* and larger *nf* generally yields better results. We train TapNet [111], an attentional prototypical network for semi-supervised learning, that achieves the lowest accuracies. We propose a benchmark for the transformer variants [13, 36, 44, 90, 101] (for details, see Appendix 6). The performance improves for all transformer variants compared to TapNet, but are notably lower than those of the convolutional and spatio-temporal methods. TST [110] with Gaussian encoding is on par with the convolutional techniques on the WD datasets. While our CNN+BiLSTM outperforms all methods on all OnHW-chars [65] datasets, it is not notably different from results achieved by the CNN+LSTM and CNN+TCN architectures, which in turn achieve the best results on the OnHW-symbols and split OnHW-equations datasets as well as on the combined cases.

Loss functions evaluation We train the CNN+BiLSTM architecture for all single-based datasets with the CCE loss as baseline and the seven variants described in Sect. 4. For FL, we search for the optimal hyperparameters for the OnHW-chars combined dataset and for the other methods for the OnHW-symbols dataset (see Appendix 6). We set $\alpha = 0.75$ and $\gamma = 8$. From the hyperparameter searches and literature recommendation, we set $\beta = 0.1$ for LSR, $\beta = 0.95$ for SBS, $\beta = 0.8$ for HBS, and $\alpha = 0.95$ for GCE. For the SCE loss, we set $\alpha = 0.5$ and $\beta = 0.5$ for the weighting of the CCE and RCE losses, respectively. Similar, the regularization terms of the JO loss are weighted by $\alpha = 1.2$ and $\beta = 0.8$. Table 7 gives an overview of the results for all loss functions for all single character-based datasets. The FL improves the CRR results of the symbols and equations datasets (WI) in comparison with the baseline, but yields worse results for the other datasets. As characters in the OnHW-chars dataset are equally distributed, the FL does not have any benefit on training performance. LSR prevents overconfidence and increases the accuracy for all datasets. LSR also achieves the highest accuracy of all losses for eight of the 12 datasets. As there are many samples that are written similarly, the model is overconfident for such samples by integrating a confidence penalty. Similar to FL, the SBS and HBS losses can only marginally improve results for symbols

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

Table 6 Recognition rates (CRR) in % for the symbols, split equations and characters WD and WI datasets

Method (\mathcal{L}_{CCE} loss)	OnHW- symbols ¹		OnHW- equations ^{1,2}		OnHW-sym. ¹ + equations ^{1,2}		OnHW-chars ³ [65]				combined	
	WD	WI	WD	WI	WD	WI	lower WD	WI	upper WD	WI	WD	WI
CNN+LSTM	96.44	80.00	95.43	84.22	95.65	85.11	88.85	79.48	92.15	85.60	78.17	68.06
CNN+BiLSTM	96.20	79.51	95.70	83.88	95.50	84.55	89.66	80.00	92.58	85.64	78.98	68.44
CNN+TCN	94.21	76.83	96.70	84.91	95.48	86.30	88.32	78.80	90.80	84.54	77.90	67.96
LSTM (2 layers)	81.18	62.85	91.05	74.11	90.64	74.70	74.76	65.63	80.46	73.86	58.88	51.41
LSTM (3 layers)	83.51	64.48	92.08	75.77	91.52	76.17	76.05	66.14	82.10	74.82	61.58	52.80
BiLSTM (2 layers)	83.30	63.01	91.39	73.43	91.48	76.60	75.80	66.28	81.88	75.50	61.19	53.60
BiLSTM (3 layers)	83.09	59.74	92.46	76.60	91.93	77.05	77.17	67.20	83.48	75.99	63.52	54.21
GRU [15]	47.57	33.22	70.80	45.73	68.36	52.96	35.12	33.98	45.69	44.90	30.72	29.22
TCN [6]	85.41	70.21	91.64	77.44	92.02	79.18	75.36	68.30	79.14	74.27	60.14	54.28
FCN [103]	92.18	74.63	94.03	81.46	94.22	82.56	81.62	71.48	85.37	77.24	67.41	58.00
RNN-FCN [40]	93.23	74.63	94.24	81.56	94.52	82.74	81.74	71.03	85.32	77.28	67.78	57.88
LSTM-FCN [39]	92.39	73.32	93.95	81.47	94.33	82.24	81.43	71.41	85.43	77.07	67.34	57.93
GRU-FCN [21]	92.39	73.32	94.29	81.18	94.49	82.05	81.71	71.57	85.26	77.30	67.22	58.10
MRNN-FCN	92.60	74.30	94.24	81.30	94.36	82.58	82.35	72.06	85.81	77.83	68.01	58.57
MLSTM- SE	89.22	70.38	93.78	82.49	94.04	82.70	79.39	71.90	85.08	77.44	69.33	60.14
FCN [40] SE, Att.	89.43	69.07	93.92	80.56	93.59	82.48	79.71	71.43	85.25	77.34	69.29	59.84
LSTM	87.74	71.85	94.12	80.13	90.14	82.10	80.21	71.26	84.68	76.69	68.63	59.25
Att.	88.37	70.54	93.95	81.18	94.14	82.78	79.97	70.92	84.57	76.71	68.76	58.84
MGRU-FCN [40]	92.60	74.30	94.21	81.28	94.43	82.25	82.17	71.90	85.81	77.92	68.22	58.79
ResCNN (64) [113]	92.23	77.41	94.58	80.95	94.55	82.07	82.52	72.00	86.91	78.64	67.55	58.67
ResNet (64) [103]	94.50	76.76	94.68	83.45	94.74	83.43	83.01	71.93	86.41	78.03	68.56	58.74
XResNet (18) [34]	93.45	74.14	94.80	81.51	94.73	82.91	81.21	69.57	86.02	76.91	66.69	56.64
XResNet (34) [34]	93.45	74.63	94.64	81.77	94.74	82.29	81.40	69.47	85.74	77.03	66.53	55.59
XResNet (50) [34]	93.66	74.47	94.63	81.74	94.83	82.76	80.99	69.14	86.05	76.69	64.98	54.38
XResNet (101) [34]	92.60	75.29	93.64	80.95	93.48	82.74	80.88	69.53	85.83	76.47	64.53	54.20
XResNet (152) [34]	92.18	73.16	93.47	80.00	92.58	81.64	80.71	69.06	85.17	76.70	64.30	53.72
XceptionTime (16) [72]	91.54	72.34	94.03	82.24	93.95	81.84	81.41	70.76	85.94	78.23	66.70	56.92
InceptionTime (32,6) [25]	91.33	76.10	94.05	81.39	93.88	82.37	80.98	72.22	85.20	78.24	66.94	58.34
InceptionTime (47,9) [25]	92.60	75.94	94.49	83.42	94.20	81.25	82.11	72.40	85.93	79.49	67.72	59.53
InceptionTime (62,9) [25]	91.97	78.07	94.83	81.57	95.01	81.74	82.15	72.76	86.05	79.81	67.89	59.62
InceptionTime (64,12) [25]	91.97	76.92	94.87	84.35	95.06	83.33	84.14	75.28	87.80	81.62	70.43	61.68
MultiIncep.Time (32,6) [25]	91.12	75.29	93.91	80.57	93.61	81.67	80.96	72.25	85.12	78.21	66.76	58.32
MiniRocket [87]	69.77	58.76	75.91	45.34	75.58	46.46	46.01	72.25	51.38	44.64	33.65	27.63
OmniScaleCNN [89]	84.78	68.09	91.76	75.46	92.23	77.49	73.70	64.13	79.54	71.23	60.58	51.88
XEM [24]	85.84	67.10	92.13	77.04	91.42	77.90	74.39	68.12	81.67	74.32	58.18	51.99
TapNet [111]	67.02	48.12	66.38	OOM	65.96	OOM	45.62	37.86	46.04	38.76	OOM	OOM
mWDN [99]	88.58	67.43	92.37	77.30	92.02	78.60	75.69	63.44	82.91	73.01	59.80	47.48
Perceiver [36]	67.40	48.10	89.60	58.10	89.30	61.10	56.20	39.70	57.08	42.89	42.72	30.28
Sinkhorn [90]	61.10	50.90	76.80	66.40	75.70	69.80	47.26	45.56	53.04	51.36	36.84	34.52
Performer [13]	55.40	47.80	76.10	68.30	74.90	66.80	47.54	46.32	53.48	51.76	36.62	34.56
Reformer [44]	56.90	47.80	75.80	70.10	75.40	70.20	47.26	47.28	53.80	51.78	35.98	34.66
Linformer [101]	53.90	42.90	75.20	67.40	74.90	68.80	48.90	44.92	53.80	51.24	34.92	34.00
TST [110] (Gaussian)	91.12	71.85	93.07	80.40	93.16	80.33	80.10	70.75	84.81	78.34	66.12	57.56
MultiTST [110]	87.53	71.19	92.36	78.82	91.96	79.46	74.19	66.59	81.81	75.18	60.81	53.95
TSiT [110]	84.99	68.09	93.30	78.98	92.91	80.28	79.56	69.90	84.55	77.21	64.81	55.73

Table 6 continued

Method (\mathcal{L}_{CCE} loss)	OnHW- symbols ¹		OnHW- equations ^{1,2}		OnHW-sym. ¹ + equations ^{1,2}		OnHW-chars ³ [65]				combined	
	WD	WI	WD	WI	WD	WI	lower WD	WI	upper WD	WI	WD	WI
CNN (from [65])	—	—	—	—	—	—	84.62	<u>76.85</u>	89.89	<u>83.01</u>	<u>70.50</u>	64.01
LSTM (from [65])	—	—	—	—	—	—	79.83	73.03	88.68	81.91	67.83	60.29
CNN+LSTM (from [65])	—	—	—	—	—	—	82.64	74.25	88.55	82.96	69.42	<u>64.13</u>
BiLSTM (from [65])	—	—	—	—	—	—	82.43	75.72	89.15	81.09	69.37	63.38

¹1-fold cross-validation split; samples interpolated to 79 timesteps. ²Split into single symbols and numbers.³5-fold cross-validation split; samples interpolated to 64 timesteps. Underlined: State-of-the-art results. **Bold**: Best results. SE: squeeze-and-excitation. Att.: attentional LSTM**Table 7** Recognition rates (CRR) in % for the symbols, split equations and characters WD and WI datasets for the CNN+BiLSTM architecture trained with different loss functions

Loss Function (CNN+BiLSTM architecture)	OnHW- symbols ¹		OnHW- equations ^{1,2}		OnHW-sym. ¹ + equations ^{1,2}		OnHW-chars ³ [65]				Combined	
	WD	WI	WD	WI	WD	WI	Lower WD	WI	Upper WD	WI	WD	WI
Categorical CE (CCE)	96.20	79.51	95.57	83.88	95.50	84.55	89.66	80.00	92.58	85.64	78.98	68.44
Focal loss (FL) [50]	95.78	79.67	95.42	84.53	95.25	85.20	88.56	78.88	91.91	85.62	77.48	68.15
Label smoothing (LSR) [67]	96.22	81.83	95.86	87.09	95.74	86.52	89.74	80.96	92.72	86.13	79.09	69.43
Boot soft (SBS) [73]	96.00	79.00	95.70	84.87	95.65	85.91	89.08	79.76	92.12	85.79	78.19	68.47
Boot hard (HBS) [73]	96.22	79.17	95.63	85.27	95.60	87.11	89.20	80.00	92.29	85.82	78.28	68.41
Generalized CE (GCE) [112]	96.44	80.83	95.81	86.46	95.64	86.69	88.18	79.34	91.51	85.49	76.91	67.76
Symmetric CE (SCE) [102]	96.44	81.00	95.76	85.15	95.58	85.43	89.24	79.90	92.09	85.84	78.11	68.65
Joint optimization (JO) [88]	97.33	82.17	95.67	85.40	95.60	85.87	89.71	80.14	92.65	86.56	79.07	69.26

¹1-fold cross-validation split; samples interpolated to 79 timesteps. ²Split into single symbols and numbers³5-fold cross-validation split; samples interpolated to 64 timesteps. **Bold**: Best results

and split equations datasets, and even decrease performances for the character datasets. HBS is slightly better than SBS. The GCE loss decreases the classification accuracy for the OnHW-chars datasets, while it achieves the second best CRR of all losses for the split OnHW-equations WD (95.81%) and WI (86.46%) datasets. Yet, the GCE loss often results in NaN loss (see Fig. 25, Appendix 7), and hence, is non-robust for our datasets. The improvement for the SCE loss is less significant than other losses and even decreases for the OnHW-chars dataset. JO leads to an improvement for all OnHW-chars datasets. JO further outperforms all losses for the WI upper task and achieves marginally lower accuracies than the LSR loss for the lower and combined datasets. LSR also achieves the highest accuracies on the OnHW-symbols WD (97.33%) and WI (82.17%) datasets. In summary, all loss variants can improve results of the CCE loss for the OnHW-symbols, split OnHW-equations and combined datasets as these are not equally distributed. LSR, SCE and JO can most significantly outperform other techniques. For more details of accuracy plots, see Appendix 7, Fig. 25.

5.3 Left-handed Writers datasets evaluation

For the left-handed writers datasets, we use the pre-trained weights from the right-handed datasets and train the CNN+BiLSTM architecture for 500 epochs. Table 8 summarizes all results for the sequence-based classification task (left) and the single character-based classification task (right). The motion dynamics of right- and left-handed writers is very different, especially with different rotations, and hence, also the sensor data are different [45]. The models can still make use of the pre-trained weights, and fine tuning leads to 1.24% CER for the OnHW-equations-L dataset for the WD task, and 15.32% CER for the OnHW-words500-L dataset, which is better than for the right-handed task. For the OnHW-wordsRandom-L dataset, the CER (5.40%) increases, while the WER (32.73%) decreases. Consistently, the results for the WI task decrease as the model overfits to specific writers due to the small amount of different left-handed writers in the training set. For single-based datasets, the fine tuning leads to a high WD classification accuracy of

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

Table 8 Evaluation results (WER, CER) in % (mean and standard deviation) for our left-handed OnHW-equations-L, OnHW-words500-L and OnHW-wordsRandom-L datasets (left), and recognition results (CRR) in % for our left-handed OnHW-symbols-L, split OnHW-equations-L and OnHW-words-L datasets (right) for the CNN+BiLSTM architecture

Dataset (CNN+BiLSTM architecture)	WD				WI				Dataset (CNN+BiLSTM architecture)	WD		WI	
	WER Mean	WER STD	CER Mean	CER STD	WER Mean	WER STD	CER Mean	CER STD		CRR Mean	CRR Mean	CRR Mean	CRR Mean
OnHW-equations-L	8.56	1.59	1.24	0.25	95.73	3.13	32.16	5.16	OnHW-symbols-L ¹	92.00	54.00		
OnHW-words500-L	47.90	17.25	15.32	6.03	97.90	1.10	81.43	11.66	OnHW-equations-L ^{1,2}	92.02	51.50		
OnHW-wordsRandom-L	32.73	3.43	5.40	1.15	99.70	0.30	72.27	15.55	OnHW- chars-L ³ [65]	Lower	94.70	–	–
									Upper	91.90	–	–	–
									Combined	82.80	–	–	–

¹1-fold cross-validation split; samples interpolated to 79 timesteps

²Split into single symbols and numbers

³5-fold cross-validation split; samples interpolated to 64 timesteps

Writer-dependent (WD) and writer-independent (WI) classification tasks

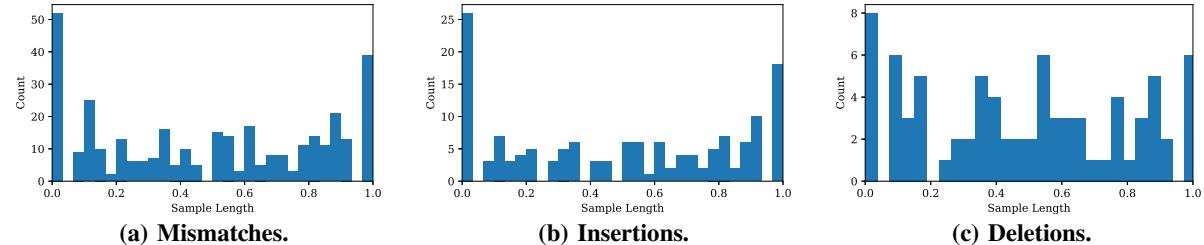


Fig. 14 Evaluation of the ED dependent on the normalized sample lengths for the OnHW-equations dataset

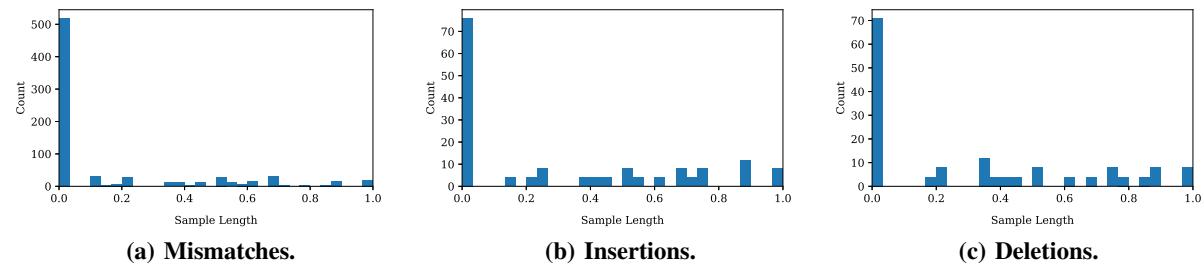


Fig. 15 Evaluation of the ED dependent on the normalized sample lengths for the OnHW-wordsTraj dataset

92% for the OnHW-symbols-L and split OnHW-equations-L datasets (compared to 96.2% and 95.57% for right-handed datasets, respectively), but decreases for WI tasks to 54% and 51.5% (compared to 79.51% and 83.88% for right-handed datasets, respectively). Due to the smaller size of the left-handed datasets, the models overfit to specific writers [45].

5.4 Edit distance and writer analysis

Evaluation of sample length-dependent edit distance We show the sample length-dependent counts of wrong predictions, i.e., mismatches, insertions and deletions, for the OnHW-equations (see Fig. 14) and OnHW-wordsTraj (see Fig. 15) datasets. For the OnHW-equations dataset, a high

appearance of mismatches and insertions appears at the starting and end characters, while deletions emerge more evenly over the whole equations. The first character of words is significantly often mismatched or has to be inserted or deleted for the OnHW-wordsTraj dataset. This shows the unequal distribution of samples for the words datasets (see Fig. 4c), while the equations dataset is very equally distributed (see Fig. 4b).

Writer-dependent evaluation Figure 16 shows the writer-dependent evaluation of the OnHW-equations dataset. The CER of many samples of several writers, e.g., ID 0, 2-4, 24-35, 42-44, and 49-53, is 0%. The CER increases only for a small number of samples. The range of the CER increases

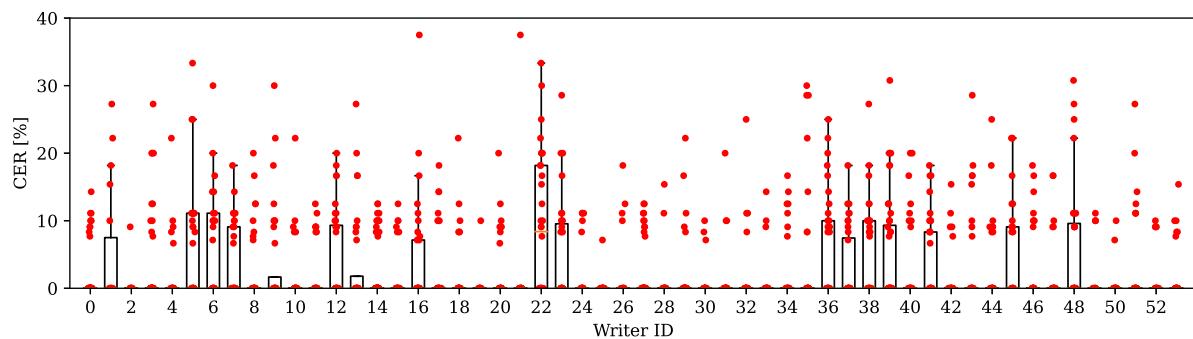


Fig. 16 Writer-dependent CER (%) for the OnHW-equations (WD) dataset

for writer IDs 1, 5-7, 22, 23, and 36-39. Hence, the writing style and with that the sensor data is different and out-of-distribution in the dataset.

without any selection participated at data recording as the handwriting style of students changes quickly with the age [7].

6 Discussion and summary

6.1 Social impact, applications and limitations

Handwriting is important in different fields, in particular *graphomotoric*. The visual feedback provided by the pen, for instance, helps young students and children to learn a new language. Hence, research for HWR is very advanced. However, state-of-the-art methods to recognize handwriting (a) require to write on a special device, which might adversely affect the writing style, (b) require to take images of the handwritten text, or (c) are based on premature technical systems, i.e., the sensor pen is only a prototype [17]. The publicly available sensor pen developed by STABILO International GmbH has previously been used by [46,65] and allows an easier data collection than previous techniques. The research for collecting devices which do not influence the handwriting style is becoming increasingly important and with it also the social impact of resulting datasets. The aim of our dataset is to support the learning of students in schools or self-paced learning from home without additional effort [4,106]. A well-known bottleneck for many machine learning algorithms is their requirement for large amounts of data samples without under-represented data patterns. For our HWR application, a large variety of different writing styles (cursive or printed characters, left- or right-handed and beginner or advanced writers), pen rotations and writing surfaces (especially different vibrations of the paper) are necessary. We provide an evaluation benchmark for right- and left-handed datasets. As motion dynamics between right- and left-handed writers are very different, extracting mutual information is a challenging task [45,63]. The ratio between both groups approximately fits the real-world distribution, i.e., the under-representation of left-handed writers (10.6%). Only adults

6.2 Experimental results

We performed several benchmarks and come to the following conclusions: (1) For the seq2seq classification task, we evaluated several methods based on CNNs in combination with RNNs on inertial-based datasets written on paper and on tablet and evaluated state-of-the-art trajectory-based datasets. Depending on the dataset size, our CNN+BiLSTM model is on par with the InceptionTime+BiLSTM architecture. A search of architecture hyperparameters is important to achieve a generalized model for a real-world application. Our transformer-based architecture could not outperform simpler convolutional models. (2) Sensor data augmentation leads to a better generalized training. (3) For the single classification task, our simple CNN+[LSTM, BiLSTM, TCN] can outperform state-of-the-art techniques. (4) Cross-entropy variants (i.e., label smoothing) improve results that are dependent on the dataset (i.e., label noise and class balance). (5) Writer-independent classification of (under-represented) left-handed writers is very challenging that is interesting for future research.

6.3 Collection consent and personal information

While recording the datasets, we collected the consent of all participants. We only collected the raw data from the sensor-enhanced pen, and for statistics the age and gender of the participant and their handedness. The handedness is necessary because the pen is differently rotated between left- and right-handed writers. The recording localization was Germany. An ID is assigned to every participant such that the dataset is fully pseudonymized. The ID is necessary for the WD and WI evaluation.

6.4 Conclusion and future research

We proposed several equations and words OnHWR datasets for a seq2seq classification task, as well as one symbol dataset for the single character classification task based on a novel sensor-enhanced pen. By utilizing (Bi)LSTM and TCN models combined with CNNs and different transformer models, we proposed a broad evaluation benchmark for lexicon-free classification. Various augmentation techniques showed notable improvement in classification accuracy. Our detailed evaluation of the WD and WI tasks sets important challenges for future research and provides a benchmark foundation for novel methodological advancements. For example, semi-supervised learning and few-shot learning such as prototypical networks could improve the classification accuracy of under-represented writers. Exploiting offline datasets for pre-training or the use of lexicon and language models might further allow the model to better learn the task.

Acknowledgements We sincerely thank all participants taking part in the data recordings and acknowledge the work of various researchers from the STABILO International GmbH, Kinemic GmbH, Fachdidaktik Deutsch Primarstufe (DID) of the Saarland University, Machine Learning and Data Analytics Lab of the Friedrich-Alexander University (FAU) and Fraunhofer Institute for Integrated Circuits (IIS) for their help with the data collection.

Funding This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A (David Rügamer) and by the research program Human-Computer-Interaction through the project “Schreibtrainer,” Grant No. 16SV8228, as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II.”

Data availability Data and materials will be publicly available upon publication that includes the OnHW-chars, OnHW-symbols, split OnHW-equations, OnHW-equations, OnHW-wordsTraj and OnHW-words500(R) datasets. We include left-handed and right-handed as well as writer-dependent (WD) and writer-independent (WI) splits. www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html

Code Availability Will be publicly available.

Declarations

Ethics approval see Sect. 6.1

Consent to participate see Sect. 6.3

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material

is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendices

In this appendix, we will give a general overview of related work in Sect. 1. We propose more details about the sensor pen in Sect. 2 and present the data acquisition and format in Sect. 3. While Sect. 4 shows additional samples, Sect. 5 proposes more detailed statistics of the datasets. We state the chosen transformer parameters in Sect. 6. Section 7 concludes with more evaluation details.

General overview of related work

Temporal convolutional networks (TCNs) TCNs consist of CNNs as encoders to extract spatio-temporal information for low-level feature computation and a classifier that captures high-level temporal information using a recurrent network. TCNs can take a series of any length and output it with the same length. They perform well in prediction tasks with time-series data. Yan et al. [107] TCNs have been used for the HWR task in [82,83].

RNNs Wigington et al. [105] proposed a CNN-LSTM model for text detection, segmentation and recognition. The performance of RNNs can be improved using dropout [68]. Carbune et al. [12] highly improve classification accuracies by a stack of **bidirectional LSTMs** [31]. Tian et al. [91] combined BiLSTMs in the word encoder with word inter-attention for a multi-task document classification approach. **Multi-dimensional RNNs** as the MDLSTM-RNNs [32] scan the input in the four possible directions, where LSTM cell inner states and output are computed from previous positions in the vertical and horizontal directions. Voigtlaender et al. [97] processed the input in a diagonal-wise fashion to enable GPU-based training and explored deeper and wider MDLSTMs architectures for HWR. Bluche [10] transformed the 2D representation into a sequence of predictions to enable end-to-end processing of paragraphs. However, these architectures are computationally expensive and extract features visually similar to CNNs; hence, 2D long-term dependencies may not be essential [70]. Dutta et al. [20] integrated a spatial transformer network into their RCNN method.

Transformers They aim for handling long-range dependencies with ease relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. Vaswani et

al. [94] showed their transformer architecture consisting of a decoder, encoder and multi-head attention to be superior in quality while being more parallelizable and requiring significantly less time to train. Kang et al. [38] introduced a novel method for offline HWR that bypasses any recurrence and uses multi-head self-attention layers at visual and textual stages. As transformer-based models scale quadratically with the sequence length due to their self-attention, the Longformer introduced an attention mechanism that scales linearly and was applied to process documents of thousands of tokens. The Performer [13] that estimates (softmax) full-rank attention transformers also use only linear complexity. The Perceiver [36] scales to high-dimensional inputs such as audio, videos, images and point-clouds by using cross-attentional principles before using a stack of transformers in the latent space.

Additional information of the sensor pen

The DigiPen by STABILO International GmbH is a sensor-enhanced ballpoint pen with internal data processing capabilities. A Bluetooth module enables live streaming of the integrated sensor at 100 Hz to a connected device. The DigiPen development kit is also publicly available.³ The pen has an ergonomic soft-touch grip zone, such that the writing feels comfortable and is as normal writing on paper. The pen's overall length is 167 mm, its diameter is 15 mm, and its weighs 25 g. The pen is equipped with a front accelerometer (STM LSM6DSL), a rear accelerometer (Freescale MMA8451Q), a gyroscope (STM LSM6DSL), a magnetometer (ALPS HSCDTD008A) and a force sensor (ALPS HSFPAR003A). The front and rear accelerometers are differently oriented. The accelerometers were adjusted to a range of $\pm 2\text{g}$ with a resolution of 16 bit of the front and 14 bit for the rear accelerometer. The gyroscope has a range of $\pm 1000^\circ/\text{s}$ (16 bit), and the magnetometer has a range of 2.4 mT (14 bit). The measurement range of the force sensor is between 0 and 5.32 N (12 bit).

Data acquisition and format

STABILO International GmbH provides a recording app to obtain the sensor data that are publicly available. Through this setup we also recorded the ground truth labels. The data were recorded over a period of 1.5 years. To achieve equally distributed datasets, we apply the following constraints. The writer has to write on a normal, white paper padded by five additional sheets, and has to sit on a chair in front of a table. The logo of the pen needs to face upwards. Users are allowed to write in a cursive or printed style. The way of holding the

pen and the size of handwriting was not constrained. Prior to recording the gyroscope and magnetometer biases and the magnetometer scaling has to be determined by calibrating the pen. We do not use the calibration data, but publish the calibration files along the datasets for possible future research. For more information, see [65].

The data format is given as following. For each dataset we will publish the raw data that consist of the calibration file, a labels file with start and end timestep, and a data file with the corresponding 13 channels for each timestep. Additionally, we already preprocess the data and upload pickle (.pkl) files. For each dataset and each of the five cross-validation splits, we generated a train and validation file with the sensor data, the corresponding label and the writer IDs.

Exemplary sensor data

Figures 17 and 18 show the sensor data of the 13 channels for an exemplary equation and words written on normal paper and on tablet. The accelerometer data are given in m/s^2 , the gyroscope data in $^\circ/\text{s}$, the magnetometer data in mT and the force sensor in N. The equation sample consists of 567 timesteps, while the word sample on paper consists of 217 timesteps and on tablet of 402 timesteps. It can be shown that for all three samples the single strokes can be clearly separated through the force sensor (see Fig. 17d). By comparing the accelerometer and gyroscope data of a selected word written on normal paper (see Fig. 18a and b) with the word written on tablet (see Fig. 18c and d), we can see that the surface of the paper introduces higher sensor noise than the surface of the tablet.

Statistics of the datasets

The characteristics of a dataset influences the behavior of a deep learning model. If the deployed context does not match the evaluation datasets, a model is unlikely to perform well. Hence, we will propose more detailed statistics of our proposed datasets, in this section, while we already compared our datasets with state-of-the-art datasets in Sect. 3.2. Table 9 proposes average and deviation timesteps and number of strokes for each sample length of the OnHW-equations, OnHW-words500 and OnHW-wordsRandom datasets. The number of timesteps per sample is significantly larger for the OnHW-equations dataset than for the words datasets. We can conclude that writing numbers and symbols requires more time as words are mostly written in cursive font, while equations are written in printed font. Hence, the deviation of timesteps is also larger for equations. The deviation in timestep lengths is important as the data have to be split by the CTC loss, and a larger deviation leads to more split varieties. Additionally, the number of strokes per sample is a significant feature for the classification task, which can be learned

³ DigiPen Development Kit: <https://stabilodigital.com/devkit-demoapp-introduction/>

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

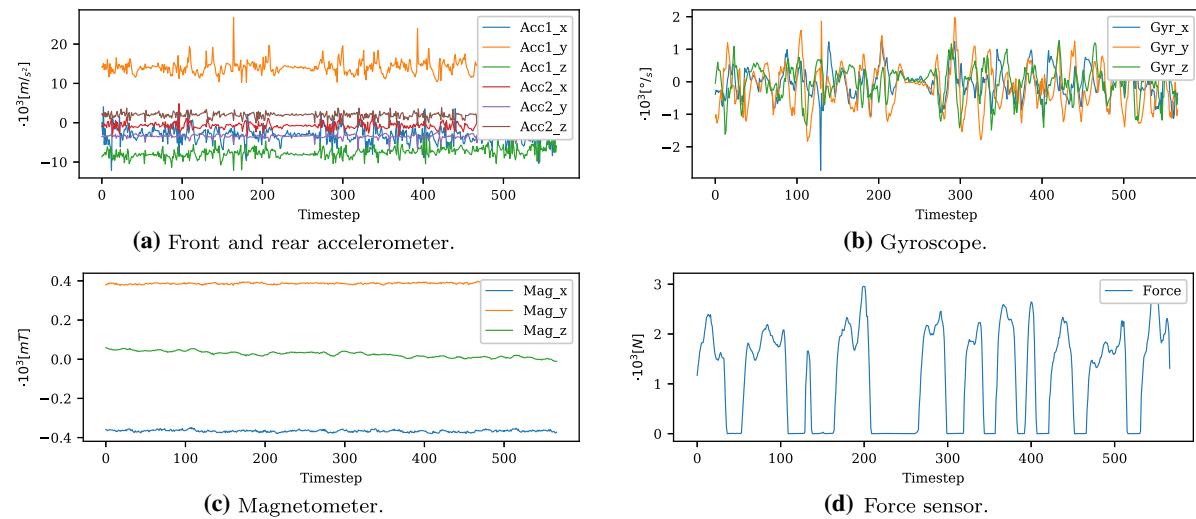


Fig. 17 Exemplary sensor data of one sample of the OnHW-equations dataset

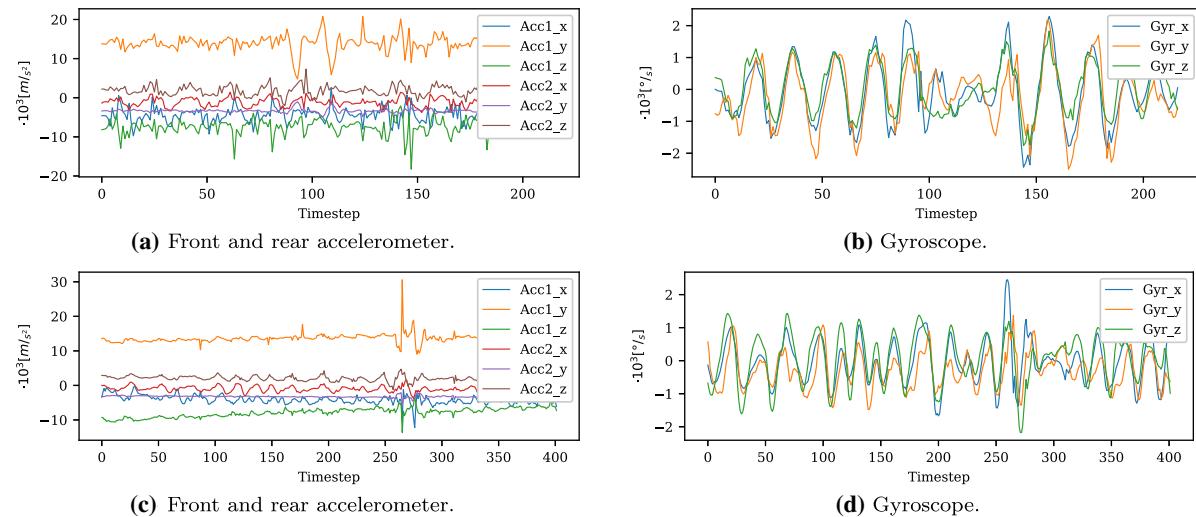


Fig. 18 Exemplary accelerometer and gyroscope data of one sample of the OnHW-words500 (top) and OnHW-wordsTraj (bottom) datasets

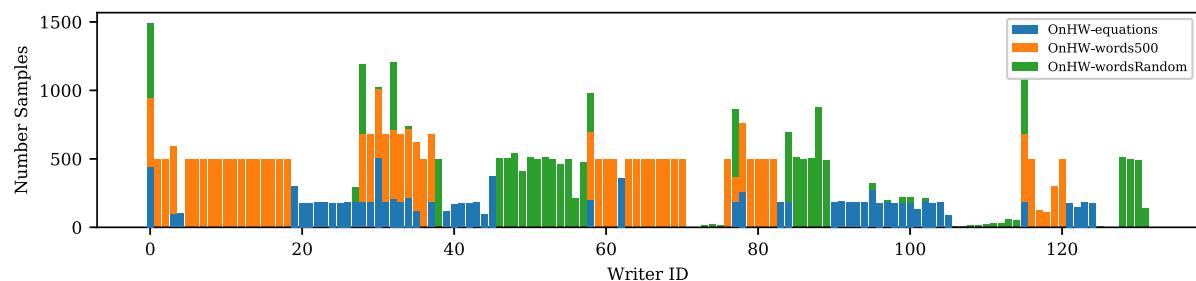
by the model from the force sensor data. The average number of strokes is clearly larger (about one to three strokes) for the OnHW-equations dataset than the words datasets, while the deviation of strokes is less. We can state from that number and symbols require many strokes for printed writing, while cursive writing of words leads to less number of strokes with a use-specific writing style. Hence, training a model for a writer-independent classification task is more difficult. To split the OnHW-equations dataset into single symbols and numbers, we use the following split constraints, where the possible number of strokes per character is 0 [1], 1 [1], 2 [1], 3 [1], 4 [1,2], 5 [2], 6 [1], 7 [1,2], 8 [1], 9 [1], + [2], - [1], · [1], : [2] and = [2].

Figure 19 gives an overview of writers contributed to the sequence-based datasets. For the OnHW-equations dataset most participants wrote about 180 equations, while for the OnHW-words500 and OnHW-wordsRandom each writer contributed about 500 words. This leads to a equally balanced dataset and a proper writer-independent evaluation.

Not only the diversity of the samples per participant is important, but also the diversity of the sensor data. In particular, out-of-distribution sensor data from one writer can decrease classification accuracy. Figure 20 gives an overview of the mean distribution per writer for the x-axis of the sensors. For the force data, the writers with IDs 7 and 17 have many outliers, while the writers with IDs 12, 37 and 42 press

Table 9 Overview of sample data length average L_A and standard deviation L_D and number of strokes average S_A and standard deviation S_D per number of labels

Dataset	2	3	4	5	6	7	8	9	10	11	12	13	14
OnHW-equations	L_A	–	–	–	228	332	420	482	559	642	703	777	855
	L_D	–	–	–	37	105	158	167	192	207	211	230	235
	S_A	–	–	–	6.21	7.90	9.28	10.37	11.76	13.05	14.21	15.57	17.03
	S_D	–	–	–	2.21	1.67	1.88	2.32	2.61	2.69	2.86	2.93	3.76
OnHW-words500	L_A	87	120	150	187	223	272	320	349	393	425	486	502
	L_D	258	36	35	51	51	63	72	95	86	90	106	118
	S_A	2.46	3.25	3.84	4.57	5.32	6.64	7.40	8.32	8.88	9.55	10.36	11.81
	S_D	1.00	1.30	1.41	1.76	1.94	2.33	2.64	2.95	2.94	3.30	3.44	4.26
OnHW-wordsRandom	L_A	111	167	201	240	290	340	397	438	493	538	598	648
	L_D	41	80	86	94	112	128	158	165	179	194	209	222
	S_A	2.83	3.89	4.64	5.25	5.95	6.82	7.69	8.50	9.23	10.08	11.11	11.74
	S_D	1.07	1.41	1.72	1.80	2.11	2.34	2.65	2.96	3.12	3.42	3.56	3.82
OnHW-equations	L_A	1,080	–	–	–	–	–	–	–	–	–	–	–
	L_D	221	–	–	–	–	–	–	–	–	–	–	–
	S_A	18.90	–	–	–	–	–	–	–	–	–	–	–
	S_D	3.15	–	–	–	–	–	–	–	–	–	–	–
OnHW-words500	L_A	609	608	–	707	712	–	–	–	–	–	–	–
	L_D	115	119	–	124	144	–	–	–	–	–	–	–
	S_A	14.41	15.27	–	16.96	15.58	–	–	–	–	–	–	–
	S_D	4.52	3.94	—	5.16	5.01	—	—	—	—	—	—	—
OnHW-wordsRandom	L_A	748	762	854	891	922	1,018	967	982	1,019	1,199	1,078	1,087
	L_D	237	233	266	260	249	250	221	209	193	206	355	143
	S_A	13.93	14.22	15.64	16.19	17.32	18.61	17.91	19.88	20.00	25.25	23.33	25.00
	S_D	4.45	4.97	5.03	5.04	5.48	5.51	6.14	7.17	5.41	6.67	5.25	2.33

**Fig. 19** Overview of writer IDs contributed to the OnHW-equations, OnHW-words500, and OnHW-wordsRandom datasets

the pen tip strongly to the paper. While the front accelerometer data are very diverse between -10^3 and 10^3 (e.g., writer 14, 25 and 45 with many outliers, against writer 16, 19 and 37 with consistent sensor data), the movement of the rear accelerometer is slower between $-4 \cdot 10^3$ and $3 \cdot 10^3$, as the pen tip typically moves faster than the rear accelerometer. The gyroscope distribution per writer draws conclusion of the writing style. These findings lead to the conclusion that the writer-dependent problem is an easier classification task than the writer-independent problem.

Transformer parameters and hyperparameter searches

Transformer parameters

This section describes the transformer parameters. For our attention-based model, we search for the optimal parameters $d_{\text{model}} = [150, 300]$, $d_k = [32, 64]$, $d_v = [32, 64]$, the number of multi-head attentions $n_{\text{head}} = [3, 4, 5]$ and a convolutional factor $c_{\text{fac}} = [4, 6, 8, 10, 16]$, while the network consists of the 1D convolutions ($c_{\text{fac}}, 2 \cdot c_{\text{fac}}, 4 \cdot c_{\text{fac}}, 8 \cdot c_{\text{fac}}$)

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

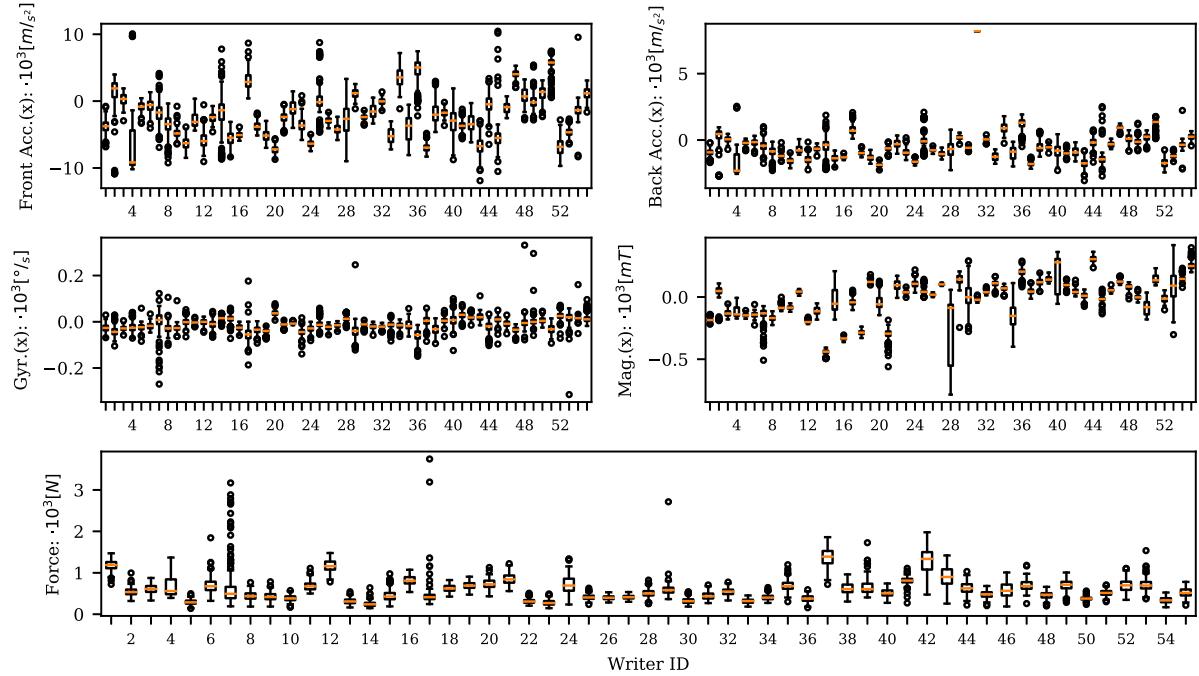


Fig. 20 Overview of mean sensor distribution per writer for the OnHW-equations dataset

and the (Bi)LSTM layers ($4 \cdot c_{\text{fac}}, 2 \cdot c_{\text{fac}}$). We train only 500 epochs, as each training takes 4h. We choose $d_{\text{model}} = 150$, $d_k = 32$, $d_v = 64$, $c_{\text{fac}} = 6$, with BiLSTM and time distribution for follow-up trainings. The number of heads n_{heads} is 3. We apply the Transformer variants Perceiver [36], Sinkhorn Transformer [90], Performer [13], Reformer [44] and Linformer [101] to the single character classification task with the following parameters. We choose non-reversible transformers without a language model or a lexicon. The input is the inertial MTS. We evaluated different combinations of last layers for all variants, i.e., with and without 1D convolution or 1D max pooling. The best results yielded a permutation with a 1D max pooling of kernel size 5 and stride 5, in combination with a linear layer of size $(\text{in}_{\text{dim}}, \text{n}_{\text{classes}})$. For the Perceiver [36], we set $\text{crossheads} = 1$, $\text{numfreq} = 4$, $\text{depth} = 2$, $\text{numlatents} = 64$, $\text{latent}_{\text{dim}, \text{heads}} = 128$, $\text{maxfreq} = 10$ and $\text{latent}_{\text{heads}} = 4$. We set $\text{attn}_{\text{drop}}$ and ff_{drop} to 0.2. $\text{n}_{\text{classes}}$ depends on the dataset and is for the OnHW-symbols and OnHW-equations dataset 15, and for the OnHW-chars dataset 26 for the lower and upper datasets and 52 for the combined dataset. We choose the parameters of the Sinkhorn Transformer [90] $\text{dim} = 1,024$, $\text{heads} = 8$, $\text{depth} = 12$, $\text{dim}_{\text{head}} = 6$ and $\text{bucket}_{\text{size}} = 20$. For the Performer [13], we choose $\text{dim} = 512$, $\text{depth} = 1$, $\text{heads} = 5$, $\text{dim}_{\text{head}} = 4$ and $\text{causal} = \text{True}$. The parameters of the Reformer [44] are $\text{dim} = 128$, $\text{heads} = 8$, $\text{bucket}_{\text{size}} = 20$, $\text{dim}_{\text{head}} = 6$, $\text{depth} = 12$, $\text{lsh}_{\text{drop}} = 0.1$ and $\text{causal} = \text{True}$.

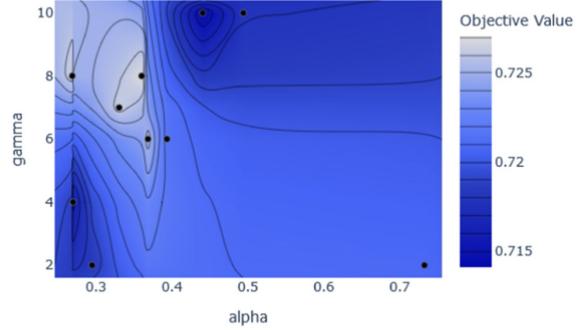


Fig. 21 Parameter search for α and γ in the Focal loss [50] with Optuna

We set for the Linformer Transformer [101] the parameters $\text{dim} = 512$, $\text{seq}_{\text{len}} = 79$ for split OnHW-equations and OnHW-symbols and $\text{seq}_{\text{len}} = 64$ for OnHW-chars [65], $\text{depth} = 12$, $\text{heads} = 5$, $\text{share}_{\text{kv}} = \text{True}$ and $\text{k} = 256$. For the Sinkhorn and Reformer Transformers, the sequence length has to be divisible by the bucket size. For Performer and Linformer Transformers, the input dimension has to be divisible by the number of heads, and hence, we exclude the magnetometer data.

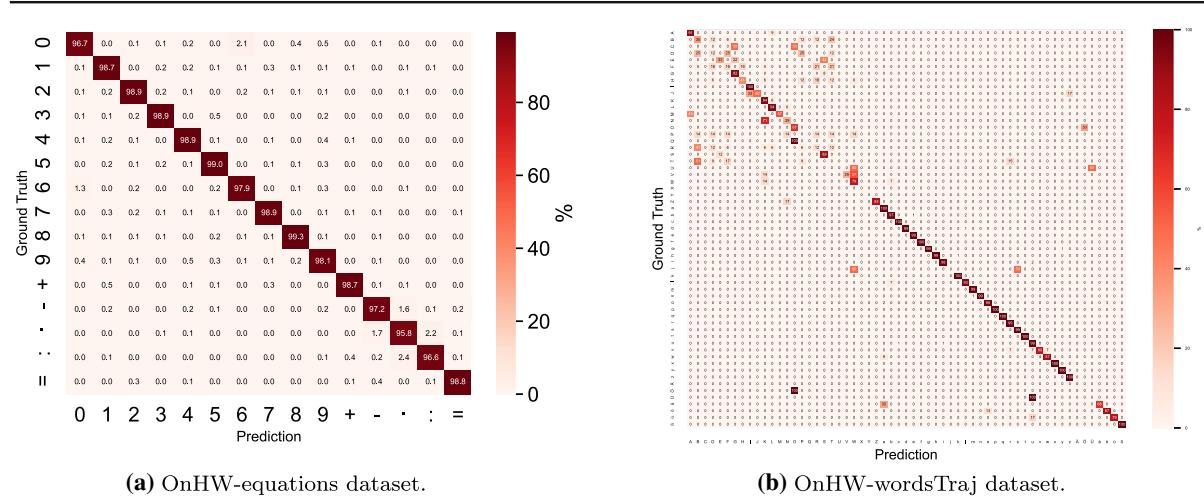


Fig. 22 Confusion matrices for mismatches

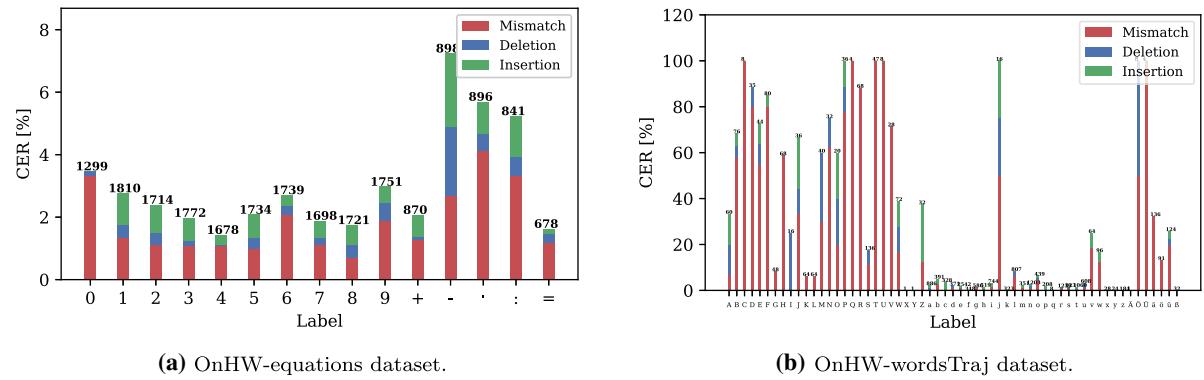


Fig. 23 Evaluation of the Edit distance with mismatch, deletion and insertion for every character prediction

Hyperparameter search

We search for the Focal loss [50] for the class balance factor $\alpha \in [0, 1]$ and $\gamma \geq 0$ in the modulating factor $(1 - p_i)^\gamma$. We use the combined OnHW-chars (WI) dataset. Figure 21 shows the hyperparameter search for α and γ with Optuna⁴. The objective value is the character recognition rate. The optimal parameters are $\alpha = 0.75$ and a large $\gamma = 8$. Note that the search space is in a small range between 71% and 73%. We use these parameters, for follow-up trainings.

Detailed evaluation

Evaluation of the accuracy per Label

Figure 22 shows confusion matrices for sequence-based classification tasks for the accuracy of predicted single class

labels regarding the ground truth class labels in %. For the OnHW-equations dataset (see Fig. 22a), the accuracies per labels are between 96.6% and 99.3%. While the ground truth '0' is confused with '6' and '9' because of the similar round shape of these numbers, the '−' is misclassified with the '·' as both symbols are short samples, and the ':' is misclassified with a single dot '·'. From analyzing the confusion matrix of the OnHW-wordsTraj (see Fig. 22b) dataset, we see two significant patterns. First, small letters are highly accurate starting from 80% (see the second part of the diagonal), while only 'j' is misclassified with 'W' and 's'. Second, capital letters are highly incorrect (see the first part of the diagonal). Letters as 'C', 'P' and 'T' are indistinguishable from other letters, while 'Q' and 'O' are interchanged. The reason is the under-representation of capital letters in the dataset (see Fig. 4c), as capital letters only appear at the starting letter of a word in German. By plotting the confusion matrix for the OnHW-wordsRandom dataset, the mismatches for capital letters improve compared to the OnHW-wordsTraj

⁴ Optuna: <https://optuna.org/>

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

Table 10 State-of-the-art evaluation results in % for the online IAM-OnDB [51], VNOnDB-words [59] and IBM-UB-1 [84] datasets

Method	IAM-OnDB [51]		VNOnDB [59]		IBM_UB_1 [84]	
	WER	CER	WER	CER	WER	CER
BiLSTM [12] ²	6.50	2.50	12.20	6.10	15.10	4.10
curve, w/o FF ¹	18.60	5.90	—	—	25.10	6.00
curve, w/ FF ¹	10.60	4.00	—	—	15.10	4.10
BiLSTM [27]	24.99	12.26	—	—	—	—
BiLSTM [31]	20.30	11.50	—	—	—	—
LSTM [52]	18.93	—	—	—	—	—
combination ²	13.84	—	—	—	—	—
BiLSTM [41] ²	26.70	8.80	—	—	22.20	6.70
Seg-and-Dec [41] ²	10.40	4.30	—	—	—	—
GoogleTask2 ⁴	—	—	19.00	6.86	—	—
IVTOVTask2 ^{3,4}	—	—	14.11	3.24	—	—
MyScriptTask2_1 ^{3,4}	—	—	2.02	1.02	—	—
MyScriptTask2_2 ^{3,4}	—	—	1.57	4.02	—	—

¹ Feature functions (FF) ² Open training set ³ VieTreeBank (VTB) corpus

⁴ Results available:here

dataset, but is still significantly higher compared with small letters. The vowel mutations ('ä', 'ö', 'ü', 'Ä', 'Ö', 'Ü') are also highly under-represented, and the classification accuracy highly decreases. Figure 23 separates the mismatches, deletions and insertions per labels (see Eq. (1)). The number on top of the box plot indicates the amount of occurrences in the validation test. For the OnHW-equations dataset (see Fig. 23a), the number '0' does not have to be inserted, and number '4' does not have to be deleted. It is significant that that the symbols with less timesteps '-' and ':' are often mismatched and missed, while only '-' has to be deleted. Numbers are distinguishable more easily. For the OnHW-wordsTraj (see Fig. 23b), again, the CER of capital letters is considerably higher than small letters. While some letters, i.e., 'C', 'Q', 'T', 'U' and 'V', are only mismatched, other edit errors appear for 'A', 'I', 'J' and 'Z'. A dataset with more capital letters could mitigate these errors.

Evaluation of sample length-dependent edit distance

We show the sample length-dependent counts of wrong predictions, i.e., mismatches, insertions and deletions, for the OnHW-equations (see Fig. 14) and OnHW-wordsTraj (see Fig. 15) datasets. For the OnHW-equations dataset, a high appearance of mismatches and insertions appears at the starting and end character, while deletions emerge more evenly over the whole equations. As previously shown, the first character of words is significantly often mismatched or has to be inserted or deleted for the OnHW-wordsTraj dataset. This shows the unequal distribution of samples for the words datasets (see Fig. 4c), while the equations dataset is very equally distributed (see Fig. 4b).

Evaluation results of state-of-the-art techniques for online HWR. This section summarizes state-of-the-art results for the IAM-OnDB [51], VNOnDB [59] and IBM-UB-1 [84] datasets (see Table 10). Graves et al. [31] (2008) started to improve the classification accuracy by proposing an alternative approach based on a RNN specifically designed for sequence labeling tasks where data contain long-range interdependencies and that is hard to segment. Liwicki et al. [52] introduced recognizers based on hidden Markov models and BiLSTMs, and on different set of features from online and offline data. Frinken et al. [27] showed that a deep BiLSTM neural network outperforms the standard BiLSTM model by combining ReLU activation with BiLSTM layers, but get a high WER of 24.99% and a CER of 12.26% on the IAM-OnDB dataset. Keysers et al. [41] used a training for feature combination, a trainable segmentation technique, unified time- and position-based input interpretation and a cascade of pruning strategies. The method achieves a WER of 26.7% and a CER of 8.80% with a BiLSTM, and up to 10.4% WER and 4.30% CER with a segmentation approach. The system is used in several Google products such as for translation. Carbune et al. [12] used bi-directional recurrent layers in combination with a softmax layer and the CTC loss. Their approach supports 102 languages. Hence, the architecture is based on a language model. The system combines methods from sequence recognition with a new input encoding using Bézier curves. This technique achieves the currently best results for the IAM-OnDB and IBM_UB_1 datasets. Feature functions (FF) introduce prior knowledge about the underlying language into the system. This method was used for the ICFHR2018 competition on Vietnamese online handwritten text recognition using VNOnDB. Along with this challenge, results from GoogleTask2, IVTOVTask2

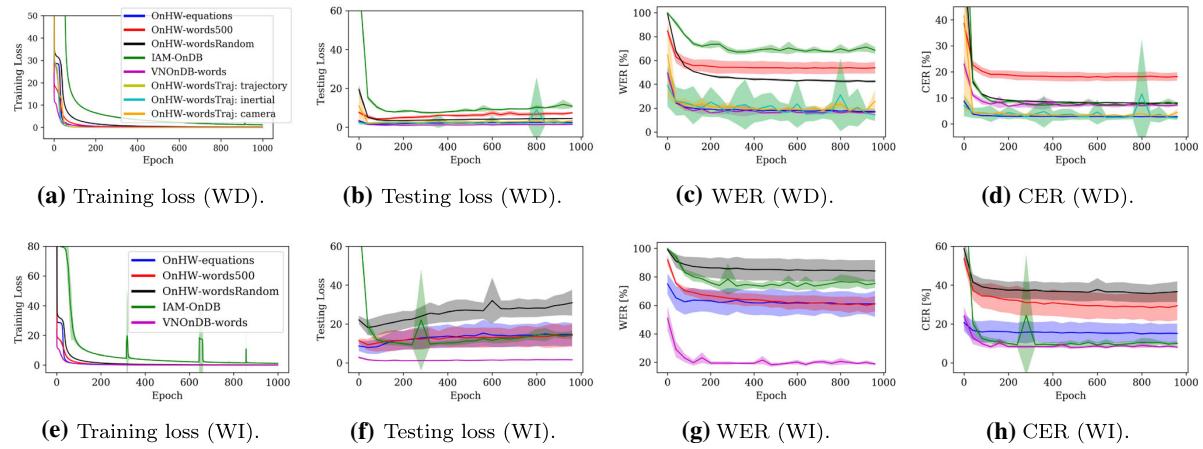


Fig. 24 Overview of training and testing losses and the evaluation metrics WER and CER in % (mean and standard deviation over fivefold cross-validation) for all WD and WI datasets

and MyScriptTask2 are available, where MyScriptTask2_2 achieves the lowest WER of 1.57% on the VNOnDB dataset. This method uses a segmentation component with a feed-forward network along with BiLSTMs and the CTC loss. The IVTOVTask2 system also uses BiLSTM layers with the CTC loss similar to our approach. Unfortunately, public code is not available for these approaches. A direct comparison of these results with our results is not possible, as we used a fivefold cross-validation of the IAM-OnDB [51] and VNOnDB-words [59] datasets, different to the train/test splits used for public results. But for our task, we can differentiate between WD and WI classification tasks. With our best model CNN+BiLSTM we achieve a CER of 6.94% (WD) and 9.11% (WI) for the IAM-OnDB dataset that is better than the BiLSTM approaches by [27,31,41], but worse than the BiLSTM by [12] and the method by [41]. On the VNOnDB-words dataset, our CER of 6.71% (WD) and the WER of 15.54% (WD) is lower than GoogleTask2, but higher than [12] and MyScriptTask2.

Error and accuracy plots

Figure 24 shows an overview of error plots for all sequence-based datasets. While the training losses converge very fast (see Fig. 24a and e) and the models slightly overfit (see Fig. 24b and f), the WER (see Fig. 24c and g) and the CER (see Fig. 24d and h) are continuously decreasing. We propose the validation accuracies (CRR) while training in Fig. 25 for single-based datasets for the eight training losses. The generalized cross-entropy (GCE) is often not robust, see Fig. 25c to j. The difference between loss function of the WD symbols and equations datasets is small (see Fig. 25a and c), but gets more important for the WI tasks (see Fig. 25b and d). From the OnHW-chars [65] dataset, we can conclude that

symmetric cross-entropy (SCE), label smoothing (LSR) and joint optimization (JO) can improve the baseline categorical cross-entropy (CCE) loss. The Focal loss (FL) [50] converges slower, and boot soft (SBS) and boot hard (HBS) are similar to CCE.

Training times

Table 11 compares training times of all methods used for our benchmark for the lower OnHW-chars [65] (WD) and our OnHW-equations (WD) datasets. For all trainings, we used Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM. TapNet [111] has the fastest training time of 3.6s, but we trained 3000 epochs for convergence. While we train our CNN+LSTM model for 1000 epochs with 8.0s each, the MLSTM-FCN [40] trains slower, but converges faster (only 200 epochs). The training times per epoch of the transformer variants [13,36,44,90,101] are significantly higher (between 15.7 and 24.5s), but the convergence is also significantly faster of less than 100 epochs. The Linformer [101] (8.8s) is as fast as our CNN+LSTM model (8.0s). For seq2seq classification tasks, our attention-based model is the fastest with 27.5s. The CNN+TCN model requires 43.5s and the CNN+LSTM model 62s, and can emphasize the advantage of attention-based models. The CNN+BiLSTM model achieves the lowest error rates, but trains clearly slower with 131s. In conclusion, transformers train faster than classical methods, but our classical CNN and RNN models achieve the highest accuracies. Modules trained with the tsai toolbox have lower training times: InceptionTime (2.0s), XceptionTime (3.8s), ResCNN (2.2s) and ResNet (2.9s). The small model FCN is very fast at training (1.5s) that increases for added temporal units such as LSTM-FCN (6.8s) and MLSTM-FCN (7.4s). The training of InceptionTime increases from 2.0s for depth

Benchmarking online sequence-to-sequence and character-based handwriting recognition from IMU...

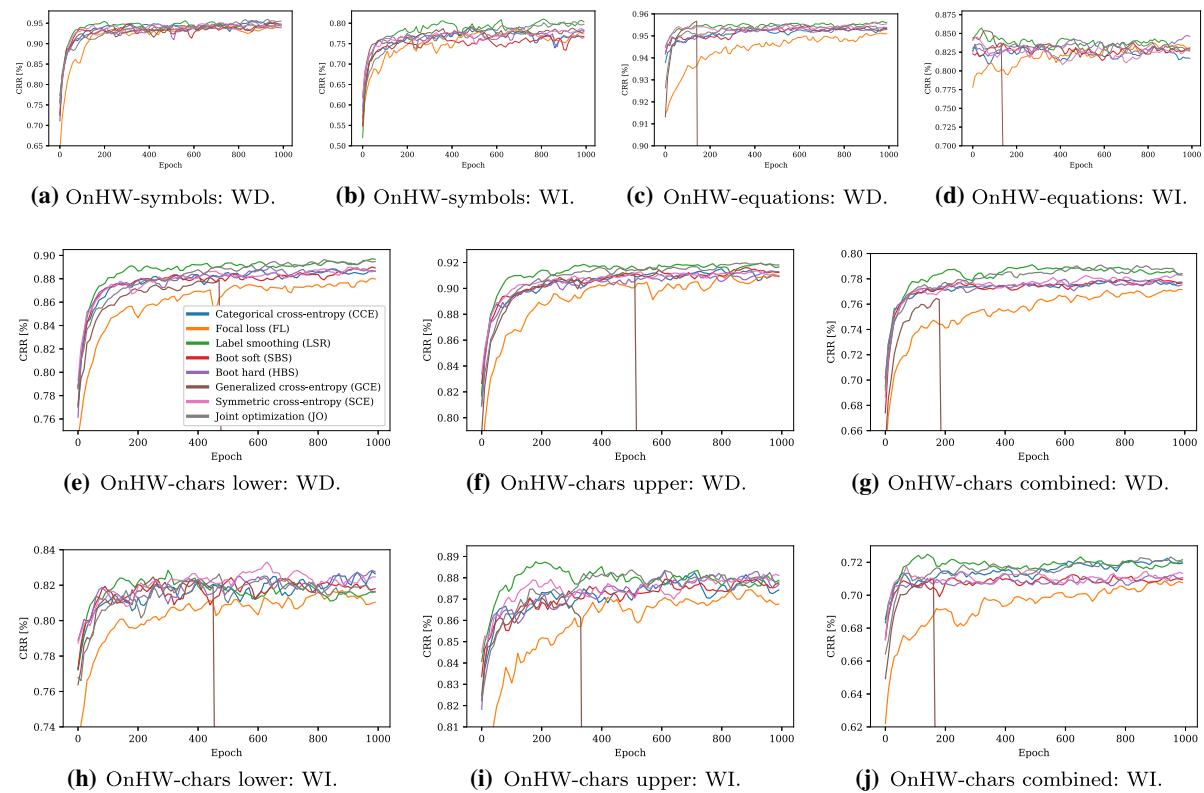


Fig. 25 Overview of validation accuracies evaluated every 10th training epoch for the WD and WI OnHW-symbols, split OnHW-equations and OnHW-chars [65] datasets

Table 11 Comparison of training times per epoch in seconds (s)

Method	OnHW-chars	OnHW-equations
CNN+LSTM	8.0	62
CNN+BiLSTM	19.7	131
CNN+TCN	7.3	43.5
Attention-based model	—	27.5
Perceiver [36]	17.1	—
Sinkhorn [90]	16.1	—
Performer [13]	15.7	—
Reformer [44]	24.5	—
Linformer [101]	8.8	—
TapNet [111]	3.6	—
MLSTM-FCN [40]	12.0	—

3 and nf 16 up to 37.6s for depth 12 and nf 128. Added BiLSTM layers up to double the training times.

References

1. Abed, H.E., Kherallah, M., Märgner, V., Alimi, A.M.: On-line Arabic handwriting recognition competition: ADAB database and participating systems. *IJDAR* **4**, 15–23 (2010). <https://doi.org/10.1109/ICDAR.2011.289>
2. Alimoglu, F., Alpaydin, E.: Combining multiple representations and classifiers for pen-based handwritten digit recognition. In: *ICDAR*, vol. 2. Ulm, Germany (1997). <https://doi.org/10.1109/ICDAR.1997.620583>
3. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Word spotting and recognition with embedded attributes. *TPAMI* **36**(12), 2552–2566 (2014). <https://doi.org/10.1109/TPAMI.2014.2339814>
4. Alonso, M.A.P.: Metacognition and sensorimotor components underlying the process of handwriting and keyboarding and their impact on learning. An analysis from the perspective of embodied psychology. *Procedia Soc. Behav. Sci.* **176**, 263–269 (2015). <https://doi.org/10.1016/j.sbspro.2015.01.470>
5. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. In: *arXiv:1607.06450* (2016)
6. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. In: *arXiv:1803.01271* (2018)
7. Barrett, P., Davies, F., Zhang, Y., Barrett, L.: The impact of classroom design on pupils' learning: final results of a holistic, Multi-level analysis. *Build. Environ.* **89**, 118–133 (2015). <https://doi.org/10.1016/j.buildenv.2015.02.013>

8. Bertolami, R., Bunke, H.: Hidden Markov model-based ensemble methods for offline handwritten text line recognition. *Pattern Recogn.* **41**(11), 3452–3460 (2008). <https://doi.org/10.1016/j.patcog.2008.04.003>
9. Bluche, T.: Deep neural networks for large vocabulary handwritten text recognition. Dissertation (2015)
10. Bluche, T.: Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. In: NIPS, pp. 838–846. Barcelona, Spain (2016)
11. Bu, Y., Xie, L., Ying, Y., Ning, C.W.J., Cao, J., Lu, S.: Handwriting-assistant: reconstructing continuous strokes with millimeter-level accuracy via attachable inertial sensors. *IMWUT* **5**(4), 1–25 (2021). <https://doi.org/10.1145/3494956>
12. Carbune, V., Gonnet, P., Deselaers, T., Rowley, H.A., Daryin, A., Calvo, M., Wang, L.L., Keyser, D., Feuz, S., Gervais, P.: Fast Multi-language LSTM-based online handwriting recognition. *IJDAR* **23**, 89–102 (2020). <https://doi.org/10.1007/s10032-020-00350-4>
13. Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., Belanger, D., Colwell, L., Weller, A.: Rethinking Attention with Performers. In: ICLR (2021)
14. Chowdhury, A., Vig, L.: An efficient end-to-end neural model for handwritten text recognition. In: BMVC (2018)
15. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: [arXiv:1412.3555](https://arxiv.org/abs/1412.3555) (2014)
16. Damerau, F.J.: A technique for computer detection and correction of spelling errors. *Commun. ACM* **7**(3), 171–176 (1964). <https://doi.org/10.1145/363958.363994>
17. Deselaers, T., Keyser, D., Hosang, J., Rowley, H.A.: GyroPen: gyroscopes for pen-input with mobile phones. *THMS* **45**(2), 263–271 (2015). <https://doi.org/10.1109/THMS.2014.2365723>
18. Doetsch, P., Kozielski, M., Ney, H.: Fast and robust training of recurrent neural networks for offline handwriting recognition. In: ICFHR, pp. 279–284 (2014). <https://doi.org/10.1109/ICFHR.2014.54>
19. Dreuw, P., Doetsch, P., Plahl, C., Ney, H.: Hierarchical hybrid MLP/HMM or rather MLP Ffatures for a discriminatively trained Gaussian HMM: A comparison for offline handwriting recognition. In: ICIP, pp. 3541–3544 (2011). <https://doi.org/10.1109/ICIP.2011.6116480>
20. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V.: Improving CNN-RNN hybrid networks for handwriting recognition. In: ICFHR, pp. 80–85 (2018). <https://doi.org/10.1109/ICFHR-2018.00023>
21. Elsayed, N., Maida, A.S., Bayoumi, M.: Deep gated recurrent and convolutional network hybrid model for univariate time series classification. In: [arXiv:1812.07683](https://arxiv.org/abs/1812.07683) (2018)
22. España-Boquera, S., Castro-Bleda, M.J., Gorbe-Moya, J., Zamora-Martínez, F.: Improving Offline handwritten text recognition with hybrid HMM/ANN models. *TPAMI* **33**(4), 767–779 (2010). <https://doi.org/10.1109/TPAMI.2010.141>
23. Fahmy, M.M.M.: Online signature verification and handwriting classification. *ASEJ* **1**(1), 59–70 (2010). <https://doi.org/10.1016/j.asej.2010.09.007>
24. Fauvel, K., Élisa Fromont, Masson, V., Faverdin, P., Termier, A.: XEM: An explainable ensemble method for multivariate time series classification. In: [arXiv:2005.03645](https://arxiv.org/abs/2005.03645) (2020)
25. Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F.: InceptionTime: finding AlexNet for Time series classification. In: [arXiv:1909.04939](https://arxiv.org/abs/1909.04939) (2019)
26. Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., Litman, R.: ScrabbleGAN: semi-supervised varying lenght handwritten text generation. In: CVPR, pp. 4324–4333 (2020). <https://doi.org/10.1109/CVPR42600.2020.00438>
27. Frinken, V., Uchida, S.: Deep BLSTM neural networks for unconstrained continuous handwritten text recognition. In: ICDAR, pp. 911–915 (2015). <https://doi.org/10.1109/ICDAR.2015.7333894>
28. Gerth, S., Klassert, A., Dolk, T., Fliesser, M., Fischer, M.H., Nottbusch, G., Festman, J.: Is handwriting performance affected by the writing surface? Comparing preschoolers', Second Graders', and adults' Writing Performance on a Tablet vs Paper. *Front. Psychol.* (2016). <https://doi.org/10.3389/fpsyg.2016.01308>
29. Graves, A.: Generating sequences with recurrent neural networks. In: [arXiv:1308.0850](https://arxiv.org/abs/1308.0850) (2014)
30. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: ICML, pp. 369–376. Pittsburgh, PA (2006). <https://doi.org/10.1145/1143844.1143891>
31. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. *TPAMI* **31**(5), 855–868 (2009). <https://doi.org/10.1109/TPAMI.2008.137>
32. Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: NIPS, pp. 545–552 (2008)
33. Guyon, I., Schomaker, L., Plamondon, R., Liberman, M., Janet, S.: UNIPEN project of on-line data exchange and recognizer benchmarks. In: ICPR, vol. 3 (1994). <https://doi.org/10.1109/ICPR.1994.576870>
34. He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., Li, M.: Bag of tricks for image classification with convolutional neural networks. In: CVPR, pp. 558–567. Long Beach, CA (2019). <https://doi.org/10.1109/CVPR.2019.00065>
35. Hussain, R., Raza, A., Siddiqi, I., Khurshid, K., Djeddi, C.: A comprehensive survey of handwritten document benchmarks: Structure, usage and evaluation. *J. Image Video Process.* (2015). <https://doi.org/10.1186/s13640-015-0102-5>
36. Jaegle, A., Gimeno, F., Brock, A., Zisserman, A., Vinyals, O., Carreira, J.: Perceiver: general perception with iterative attention. In: ICML (2021)
37. Kaitiy, M., Balakrishnan, V.: An integrated semi-automated framework for domain-based polarity words extraction from an unannotated non-English corpus. *J. Supercomput.* **76**, 9772–9799 (2020). <https://doi.org/10.1007/s11227-020-03222-0>
38. Kang, L., Riba, P., Rusinol, M., Fornes, A., Villegas, M.: Pay attention to what you read: non-recurrent handwritten text-line recognition. In: [arXiv:2005.13044](https://arxiv.org/abs/2005.13044) (2020)
39. Karim, F., Majumdar, S., Darabi, H., Chen, S.: LSTM fully convolutional networks for time series classification. In: [arXiv:1709.05206](https://arxiv.org/abs/1709.05206) (2017)
40. Karim, F., Majumdar, S., Darabi, H., Harford, S.: Multivariate LSTM-FCNs for time series classification. *Neural Netw.* **116**, 237–245 (2019). <https://doi.org/10.1016/j.neunet.2019.04.014>
41. Keyser, D., Deselaers, T., Rowley, H.A., Wang, L.L., Carbune, V.: Multi-language online handwriting recognition. *TPAMI* **36**(6), 1180–1194 (2017). <https://doi.org/10.1109/TPAMI.2016.2572693>
42. Kherallah, M., Elbaati, A., Abed, H.E., Alimi, A.M.: The On/Off (LMCA) Dual Arabic handwriting database. In: ICFHR (2008)
43. Kim, S., Hori, T., Watanabe, S.: Joint CTC-attention based end-to-end speech recognition using multi-task learning. In: [arXiv:1609.06773](https://arxiv.org/abs/1609.06773) (2017)
44. Kitaev, N., Kaiser, L., Levskaya, A.: Reformer: the efficient transformer. In: ICLR (2020)
45. Klaß, A., Lorenz, S.M., Lauer-Schmalz, M.W., Rügamer, D., Bischl, B., Mutschler, C., Ott, F.: Uncertainty-aware evaluation of time-series classification for online handwriting recognition with domain shift. In: IJCAI-ECAI Workshop on Spatio-Temporal

Reasoning and Learning (STRL), vol. 3190. Vienna, Austria (2022)

46. Koellner, C., Kurz, M., Sonnleitner, E.: What did you mean? An evaluation of online character recognition approaches. In: WiMob, pp. 1–6. Barcelona, Spain (2019). <https://doi.org/10.1109/WiMOB.2019.8923384>
47. Kowsari, K., Meimandi, K.J., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D.: Text classification algorithms: a survey. In: Information, vol. 10(4). Switzerland (2019). <https://doi.org/10.3390/info10040150>
48. Lewenstein, W.I.: Binary codes capable of correcting deletions, insertions, and reversals. *Dokl. Akad. Nauk. SSSR* **163**(4), 845–848 (1965)
49. Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P.S., He, L.: A survey on text classification: from shallow to deep learning. In: *arXiv:arXiv:2008.00364* (2020)
50. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV, pp. 2980–2988 (2017). <https://doi.org/10.1109/ICCV.2017.324>
51. Liwicki, M., Bunke, H.: IAM-OnDB - an On-Line English sentence database acquired from handwritten text on a whiteboard. In: ICDAR, pp. 956–961. Seoul, Korea (2005). <https://doi.org/10.1109/ICDAR.2005.132>
52. Liwicki, M., Bunke, H., Pittman, J.A., Knerr, S.: Combining diverse systems for handwritten text line recognition. *Mach. Vis. Appl.* **22**(1), 39–51 (2011). <https://doi.org/10.1016/j.patcog.2008.10.030>
53. Long Ma, L., dan Liu, H., Wu, J.: MRG-OHTC database for online handwritten Tibetan character recognition. In: ICDAR, pp. 207–211. Beijing, China (2011). <https://doi.org/10.1109/ICDAR.2011.50>
54. Michael, J., Labahn, R., Grüning, T., Zöllner, J.: Evaluating sequence-to-sequence models for handwritten text recognition. In: ICDAR (2019). <https://doi.org/10.1109/ICDAR.2019.00208>
55. Mouchère, H., Viard-Gaudin, C., Zanibbi, R., Garain, U., Kim, D.H., Kim, J.H.: ICDAR 2013 CROHME: third international competition on recognition of online handwritten mathematical expressions. In: ICDAR. Washington, DC (2013). <https://doi.org/10.1109/ICDAR.2013.288>
56. Nakagawa, M., Higashiyama, T., Yamanaka, Y., Sawada, S., Higashigawa, L., Akiyama, K.: On-line handwritten character pattern database sampled in a sequence of sentences without any writing instructions. In: ICDAR, vol. 1, pp. 376–381. Ulm, Germany (1997). <https://doi.org/10.1109/ICDAR.1997.619874>
57. Nakagawa, M., Matsumoto, K.: Collection of on-line handwritten Japanese character pattern databases and their analysis. *IJDAR* **7**, 69–81 (2004). <https://doi.org/10.1007/s10032-004-0125-4>
58. Nguyen, H.T., Nguyen, C.T., Bao, P.T., Nakagawa, M.: A database of unconstrained Vietnamese online handwriting and recognition experiments by recurrent neural networks. *Pattern Recogn.* **78**, 291–306 (2018). <https://doi.org/10.1016/j.patcog.2018.01.013>
59. Nguyen, H.T., Nguyen, C.T., Nakagawa, M.: ICFHR 2018 - competition on vietnamese online handwritten text recognition using HANDS-VNOnDB (VOHTR2018). In: ICFHR, pp. 494–499. Niagara Falls, NY (2018). <https://doi.org/10.1109/ICFHR-2018.2018.00092>
60. Ofitserov, E., Tsvetkov, V., Nazarov, V.: Soft edit distance for differentiable comparison of symbolic sequences. In: *arXiv:1904.12562* (2019)
61. Oguiza, I.: tsai - a state-of-the-art deep learning library for time series and sequential data. Github (2020). <https://github.com/timeseriesAI/tsai>
62. Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C.: Cross-modal common representation learning with triplet loss functions. In: *arXiv:2202.07901* (2022)
63. Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C.: Domain adaptation for time-series classification to mitigate covariate shift. In: ACMMM (2022). <https://doi.org/10.1145/3503161.3548167>
64. Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C.: Joint classification and trajectory regression of online handwriting using a multi-task learning approach. In: WACV, pp. 266–276. Waikoloa, HI (2022). <https://doi.org/10.1109/WACV51458.2022.00131>
65. Ott, F., Wehbi, M., Hamann, T., Barth, J., Eskofier, B., Mutschler, C.: The OnHW Dataset: Online Handwriting Recognition from IMU-enhanced ballpoint pens with machine learning. In: IMWUT, vol. 4(3), Article 92. Cancún, Mexico (2020). <https://doi.org/10.1145/3411842>
66. Peng, D., Xie, C., Li, H., Jin, L., Xie, Z., Ding, K., Huang, Y., Wu, Y.: Towards fast, accurate and compact online handwritten Chinese text recognition. In: ICDAR, pp. 157–171 (2021). https://doi.org/10.1007/978-3-030-86334-0_11
67. Pereyra, G., Tucker, G., Chorowski, J., Kaiser, L., Hinton, G.: Regularizing neural networks by penalizing confident output distributions. In: ICLR Workshop (2017)
68. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: ICFHR, pp. 285–290 (2014). <https://doi.org/10.1109/ICFHR.2014.55>
69. Plamondon, R., Srihari, S.N.: On-line and off-line handwriting recognition: a comprehensive survey. *TPAMI* **22**(1), 63–84 (2000). <https://doi.org/10.1109/34.824821>
70. Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition?. In: ICDAR, pp. 67–72 (2017). <https://doi.org/10.1109/ICDAR.2017.20>
71. Quiniou, S., Anquetil, E., Carbonnel, S.: Statistical language models for on-line handwritten sentence recognition. *ICDAR* **1**, 516–520 (2005). <https://doi.org/10.1109/ICDAR.2005.220>
72. Rahimian, E., Zabihi, S., Atashzad, S.F., Asif, A., Mohammadi, A.: XceptionTime: a novel deep architecture based on depth-wise separable convolutions for hand gesture classification. In: *arXiv:1911.03803* (2019)
73. Reed, S.E., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A.: Training deep neural networks on noisy labels with bootstrapping. In: ICLR Workshop (2015)
74. Reimers, N., Gurevych, I.: Optimal hyperparameters for deep LSTM-networks for sequence labeling tasks. In: EMNLP, pp. 338–348. Copenhagen, Denmark (2017)
75. Rijhwani, S., Anastasopoulou, A., Neubig, G.: OCR post correction for endangered language texts. In: EMNLP, pp. 5931–5942 (2020). <https://doi.org/10.18653/v1/2020.emnlp-main.478>
76. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: MICCAI, Springer, LNCS, vol. 9351, pp. 234–241 (2015). https://doi.org/10.1007/978-3-319-24574-4_28
77. Scheidl, H., Fiel, S., Sablatnig, R.: Word beam search: a connectionist temporal classification decoding algorithm. In: ICFHR, pp. 253–258. Niagara Falls, NY (2018). <https://doi.org/10.1109/ICFHR-2018.2018.00052>
78. Schomaker, L.: The ICDAR 2003 informal competition for the recognition of on-line words: the Unipen-ICROW-03 Benchmark Set. In: <https://www.ai.rug.nl/lambert/unipen/icdar-03-competition/> (2003)
79. Schrapel, M., Stadler, M.L., Rohs, M.: Pentelligence: combining pen tip motion and writing sounds for handwritten digit recognition. *Conf. Hum. Factors Comput. Syst.* **131**, 1–11 (2018). <https://doi.org/10.1145/3173574.3173705>

80. Seni, G., Kripásundar, V., Srihari, R.K.: Generalizing edit distance to incorporate domain information: handwritten text recognition as a case study. *Pattern Recogn.* **29**(3), 405–414 (1996). [https://doi.org/10.1016/0031-3203\(95\)00102-6](https://doi.org/10.1016/0031-3203(95)00102-6)

81. Seni, G., Srihari, R.K., Nasrabadi, N.: Large vocabulary recognition of on-line handwritten cursive words. *TPAMI* **18**(7), 757–762 (1996). <https://doi.org/10.1109/34.506798>

82. Sharma, A., Ambati, R., Jayagopi, D.B.: Towards faster offline handwriting recognition using temporal convolutional networks. In: NCVPRIPG, pp. 344–354 (2020). <https://doi.org/10.1109/ACOMP.2019.90015>

83. Sharma, A., Jayagopi, D.B.: Towards efficient unconstrained handwriting recognition using dilated temporal convolutional network. *Expert Syst. Appl.* (2021). <https://doi.org/10.1016/j.eswa.2020.114004>

84. Shivram, A., Ramaiah, C., Setlur, S., Govindaraju, V.: IBM_UB_1: a dual mode unconstrained english handwriting dataset. In: ICDAR, pp. 13–17 (2013). <https://doi.org/10.1109/ICDAR.2013.12>

85. Sudholt, S., Fink, G.A.: Attribute CNNs for word spotting in handwritten documents. *IJDAR* **21**, 199–218 (2018). <https://doi.org/10.1007/s10032-018-0295-0>

86. Synnaeve, G., Xu, Q., Kahn, J., Likhomanenko, T., Grave, E., Pratap, V., Sriram, A., Liptchinsky, V., Collobert, R.: End-to-End ASR: from supervised to semi-supervised learning with modern architectures. In: ICML Workshop, Vienna, Austria (2020)

87. Tan, C.W., Dempster, A., Bergmeir, C., Webb, G.I.: MultiRocket: multiple pooling operators and transformations for fast and effective time series classification. In: [arXiv:2102.00457](https://arxiv.org/abs/2102.00457) (2021)

88. Tanaka, D., Ikami, D., Yamasaki, T., Aizawa, K.: Joint optimization framework for learning with noisy labels. In: CVPR, pp. 5552–5560. Salt Lake City, UT (2018). <https://doi.org/10.1109/CVPR.2018.00582>

89. Tang, W., Long, G., Liu, L., Zhou, T., Jiang, J., Blumenstein, M.: Rethinking 1D-CNN for time series classification: a stronger baseline. In: [arXiv:2002.10061](https://arxiv.org/abs/2002.10061) (2020)

90. Tay, Y., Bahri, D., Yang, L., Metzler, D., Juan, D.C.: Sparse Sinkhorn attention. In: [arXiv:2002.11296](https://arxiv.org/abs/2002.11296) (2020)

91. Tian, B., Zhang, Y., Wang, J., Xing, C.: Hierarchical inter-attention network for document classification with multi-task learning. In: IJCAI, pp. 3569–3575 (2019). <https://doi.org/10.24963/ijcai.2019/495>

92. Uhang, J., Du, J., Yang, Y., Song, Y.Z., Dai, L.: SRD: a tree structure based decoder for online handwritten mathematical expression recognition. *Trans. Multimed.* **23**, 2471–2480 (2020)

93. Um, T.T., Pfister, F.M.J., Pichler, D., Endo, S., Lang, M., Hirche, S., Fietzek, U., Kulic, D.: Data augmentation of wearable sensor data for Parkinson’s disease monitoring using convolutional neural networks. In: ICMI, pp. 216–220. Glasgow, UK (2017). <https://doi.org/10.1145/3136755.3136817>

94. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.: Attention is all you need. In: NIPS, pp. 5998–6008. Long Beach, CA (2017)

95. Viard-Gaudin, C., Lallican, P.M., Binter, P., Knerr, S.: The IRESTE On/Off (IRONOFF) dual handwriting database. In: ICDAR, pp. 455–458 (1999). <https://doi.org/10.1109/ICDAR.1999.791823>

96. Vinciarelli, A., Perrone, M.P.: Combining online and offline handwriting recognition. In: ICDAR, pp. 844–848. Edinburgh, UK (2003). <https://doi.org/10.1109/ICDAR.2003.1227781>

97. Voigtlaender, P., Doetsch, P., Ney, H.: Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In: ICFHR, pp. 228–233 (2016). <https://doi.org/10.1109/ICFHR.2016.0052>

98. Wang, D.H., Liu, C.L., Zhou, X.D.: An approach for real-time recognition of online Chinese handwritten sentences. *Pattern Recogn.* **45**(10), 3661–3675 (2012). <https://doi.org/10.1016/j.patcog.2012.04.020>

99. Wang, J., Wang, Z., Li, J., Wu, J.: A transformer-based framework for multivariate time series representation learning. In: SIGKDD, pp. 2437–2446 (2018). <https://doi.org/10.1145/3219819.3220060>

100. Wang, J.S., Hsu, Y.L., Chu, C.L.: Online handwriting recognition using an accelerometer-based pen device. In: CSE (2013). <https://doi.org/10.2991/cse.2013.52>

101. Wang, S., Li, B.Z., Khabsa, M., Fang, H., Ma, H.: Linformer: self-attention with linear complexity. In: [arXiv:2006.04768](https://arxiv.org/abs/2006.04768) (2020)

102. Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., Bailey, J.: Symmetric cross entropy for robust learning with noisy labels. In: ICCV, pp. 322–330. Seoul, Korea (South) (2019). <https://doi.org/10.1109/ICCV.2019.00041>

103. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: a strong baseline. In: [arXiv:1611.06455](https://arxiv.org/abs/1611.06455) (2016)

104. Webbi, M., Hamann, T., Barth, J., Kämpf, P., Zanca, D., Eskofier, B.: Towards an IMU-based pen online handwriting recognizer. In: ICDAR, pp. 289–303 (2021)

105. Wigington, C., Tensmeyer, C., Davis, B., Barrett, W., Price, B., Cohen, S.: Start, follow, read: end-to-end full-page handwriting recognition. In: ECCV, pp. 372–388 (2018). https://doi.org/10.1007/978-3-030-01231-1_23

106. Wiley, R.W., Rapp, B.: The effects of handwriting experience of literacy learning. *Psychol. Sci.* **32**(7), 1086–1103 (2021). <https://doi.org/10.1177/0956797621993111>

107. Yan, J., Mu, L., Wang, L., Ranjan, R., Zomaya, A.Y.: Temporal convolutional networks for the advance prediction of ENSO. *Nat. Sci. Rep.* (2020). <https://doi.org/10.1038/s41598-020-65070-5>

108. Yana, B., Onoye, T.: Fusion networks for air-writing recognition. In: MMM, pp. 142–152 (2018). https://doi.org/10.1007/978-3-319-73600-6_13

109. Yousef, M., Bishop, T.E.: OrigamiNet: weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. In: CVPR, pp. 14710–14719. Seattle, WA (2020). <https://doi.org/10.1109/CVPR42600.2020.01472>

110. Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C.: A transformer-based framework for multivariate time series representation learning. In: SIGKDD, pp. 2114–2124 (2021). <https://doi.org/10.1145/3447548.3467401>

111. Zhang, X., Gao, Y., Lin, J., Lu, C.T.: TapNet: multivariate time series classification with attentional prototypical network. In: AAAI, pp. 6845–6852 (2020). <https://doi.org/10.1609/aaai.v34i04.6165>

112. Zhang, Z., Sabuncu, M.R.: Generalized cross entropy loss for training deep neural networks with noisy labels. In: NIPS, pp. 8778–8788. Montréal, Canada (2018)

113. Zou, X., Wang, Z., Li, Q., Sheng, W.: Integration of residual network and convolutional neural network along with various activation functions and global pooling for time series classification. *Neurocomputing* **367**, 39–45 (2019)

Chapter 7

Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

Contributing Article

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In *Proceedings of the ACM International Conference on Multimedia (ACMMM)*, pages 5934–5943, Lisboa, Portugal, October 2022. doi:10.1145/3503161.3548167.

Publisher Website

<https://dl.acm.org/doi/10.1145/3503161.3548167>

Copyright License

This work is licensed under a Creative Commons Attribution International 4.0 License (<https://creativecommons.org/licenses/by/4.0/>). © Copyright held by the owner/author(s).

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing, visualization, and project administration. David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Lucas Heublein contributed for the methodology (i.e., creation

of models), software, validation, investigation, and data curation (i.e., data management). Bernd Bischl contributed by supervision. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), visualization (i.e., presentation of the published work), supervision, and funding acquisition of the project “Schreibtrainer” (grant number 16SV8228) by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”. This paper was presented by Felix Ott as a poster at the LMU Munich Summer Retreat in July 2022 in Herrsching am Ammersee, Germany. Furthermore, this paper was presented as a video and a poster at the ACM International Conference on Multimedia (ACMMM) in October 2022 in Lisbon, Portugal. This paper together with experiments from Chapter 2 was presented by Felix Ott as a poster at the Bavarian International Conference on Artificial Intelligence (AI.BAY) in February 2023 in Munich, Germany, and was awarded 1st place in the best poster category. Additionally, Chapter 2 and the contributing paper was presented at the Workshop on Optimization and Machine Learning by the ADA Lovelace Center for Analytics, Data and Applications, titled “Domain Adaptation for Time-Series Classification: A Benchmark and Method”, in March 2023 in Waischenfeld, Germany.

Datasets and Source Code

This paper uses the OnHW-chars, OnHW-symbols, and split OnHW-equations datasets from right-handed writers and left-handed writers. The synthetically generated sinus curves can be reproduced with the published source code. The source code to reproduce evaluation results is publicly available at:

<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>

Statement about Recent, Related Research¹⁷

The paper by Singhal et al. (2023) incorporated the contributing paper in their DA overview. The Match-and-Deform (MAD) approach proposed by Tavenard et al. (2022) aims to establish correspondences between time-series in the source and target domains while accounting for temporal distortions. The optimization problem is constructed based on an optimal transport loss and a dynamic time warping (DTW) loss, which enables simultaneous alignment of the time-series. In contrast to our approach presented in Ott et al. (2022a) where we first solve the transformation problem step-wise and then select the optimal transformation by aligning statistical properties, the MAD approach jointly learns both these steps. The step-wise approach provides a more comprehensive assessment of the impact of individual steps. Additionally, the combination of loss functions necessitates the tuning of hyperparameters to achieve a balance between both losses. Although the optimal transport loss employed by Tavenard et al. (2022) is based on the p -Wasserstein distance with $p = 2$, we conducted experiments using the Earth’s Movers distance (EMD) with and without Sinkhorn regularization. Instead of temporally aligning the features of

embeddings using DTW, our proposed method aligns statistical properties of embeddings using various metrics such as Cosine similarity, Pearson correlation, MMD, CORAL, or HoMM.

Shi et al. (2022) presented a comprehensive survey on unsupervised DA for human activity recognition using time-series data, with a focus on MMD and CORAL for feature alignment. Recently, a benchmarking suite named AdaTime (Ragab et al., 2023) has been introduced for DA on time-series data, which facilitates the comparison of various methods including MMD, Deep CORAL, HoMM, MMDA (a combination of MMD and CORAL), DANN, CDAN, among others. Since the AdaTime toolbox was published after the publication of the contributing paper (Ott et al., 2022a), we conducted experiments on the OnHW datasets using AdaTime in Chapter 2.

Alipour & Tahmoresnezhad (2021) proposed the statistical distribution alignment and progressive pseudo label selection (SDA-PPLS) method that learns two projection matrices for the source and target domains. The primary objective is to map both domains into a latent subspace with a shared feature space. In addition to the first-order distribution matching technique (i.e., MMD), the SDA-PPLS method employs a second-order variant known as maximum covariance discrepancy (MCD). Ozyurt et al. (2023) introduced CLUDA, a nearest-neighbor contrastive learning framework for learning contextual representations in MTS between features of source and target domain embeddings, along with a discriminator network. However, as this model relies on five distinct loss functions, it necessitates an extensive hyperparameter search. Comparing the evaluation results of CLUDA with those of HoMM, MMDA, CAN, CDAN, Deep CORAL, and DSAN, it can be observed that CLUDA outperforms the state-of-the-art techniques on medical MTS data. The source code is currently unavailable.

Raincoat (He et al., 2023) aims to tackle both closed-set and universal unsupervised DA on time-series data by leveraging both frequency and temporal features. The model aligns features across domains while also correcting for misalignments. They employ the Sinkhorn divergence as a suitable divergence measurement for aligning the source and target domains, which is consistent with the results presented in the contributing paper (Ott et al., 2022a). Moreover, He et al. (2023) demonstrate that MMD has a theoretical weakness in terms of vanishing gradients. The implementation of their proposed method is based on AdaTime (Ragab et al., 2023). Raincoat was evaluated on several time-series datasets, namely HAR, HHAR, WISDM, SSC (EEG), and Boiler datasets. However, they only evaluated a subset of the source-target scenarios. In contrast, in Chapter 2 of our study, we evaluated all possible scenarios. Raincoat outperforms several state-of-the-art methods, including CoDATS (Wilson et al., 2020), AdvSKM (Liu & Xe, 2021), DIRT-T (Shu et al., 2018), CDAN (Long et al., 2018), and Deep CORAL (Sun et al., 2016).

The proposed DA network CDAN (Ma et al., 2022) distinguishes between the differences in the internal dependence structure and those in the marginals. To achieve this, CDAN calculates the copula distance between the marginals of the source and target domain features' distributions. Since CDAN considers both the marginal and copula feature differences (i.e., as a sum), the model provides a more effective approach for detecting changes in the marginal distributions and dependence structure. While this approach could serve

as an alternative distance measure in the contributing paper (Ott et al., 2022a), the source code is currently unavailable.

The paper by Eldele et al. (2023) provides a comprehensive survey of label-efficient representation learning techniques for time-series data. In the context of offline HWR, Kohút & Hradiš (2023) demonstrate that training a model with CTC and fine-tuning with data augmentation is effective for DA, particularly for small target domain datasets and writer-dependent and writer-independent tasks. This technique is resistant to overfitting. Kohút et al. (2023) propose a DA approach that adapts to a new writer using a writer style block and an adaptive instance normalization layer, which learns dedicated writer parameters conditioned on learned embeddings of the partitions. However, the fine-tuning-based DA approach (Kohút & Hradiš, 2023) outperforms this method.

Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

Felix Ott

Fraunhofer IIS, Fraunhofer Institute
for Integrated Circuits
Nürnberg, Germany
LMU Munich
felix.ott@iis.fraunhofer.de

David Rügamer

LMU Munich
Munich, Germany
RWTH Aachen
david.ruegamer@stat.uni-
muenchen.de

Lucas Heublein

Fraunhofer IIS, Fraunhofer Institute
for Integrated Circuits
Nürnberg, Germany
heublels@iis.fraunhofer.de

Bernd Bischl

LMU Munich
Munich, Germany
bernd.bischl@stat.uni-muenchen.de

Christopher Mutschler

Fraunhofer IIS, Fraunhofer Institute
for Integrated Circuits
Nürnberg, Germany
christopher.mutschler@iis.fraunhofer.de

ABSTRACT

The performance of a machine learning model degrades when it is applied to data from a similar but different domain than the data it has initially been trained on. To mitigate this domain shift problem, domain adaptation (DA) techniques search for an optimal transformation that converts the (current) input data from a source domain to a target domain to learn a domain-invariant representation that reduces domain discrepancy. This paper proposes a novel supervised DA based on two steps. First, we search for an optimal class-dependent transformation from the source to the target domain from a few samples. We consider optimal transport methods such as the earth mover's distance, Sinkhorn transport and correlation alignment. Second, we use embedding similarity techniques to select the corresponding transformation at inference. We use correlation metrics and higher-order moment matching techniques. We conduct an extensive evaluation on time-series datasets with domain shift including simulated and various online handwriting datasets to demonstrate the performance.

CCS CONCEPTS

- Computing methodologies → Learning under covariate shift; Learning latent representations.

KEYWORDS

Domain adaptation, domain shift, optimal transport, embedding similarity, time-series classification, online handwriting recognition

ACM Reference Format:

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), Oct. 10–14, 2022, Lisboa, Portugal*. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3503161.3548167>



This work is licensed under a Creative Commons Attribution International 4.0 License.

MM '22, October 10–14, 2022, Lisboa, Portugal

© 2022 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-9203-7/22/10.

<https://doi.org/10.1145/3503161.3548167>

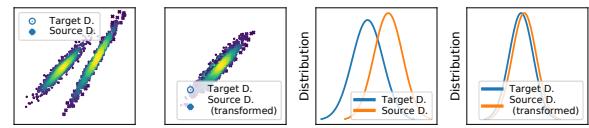


Figure 1: Domain adaptation. To compensate the domain shift the source data is transformed into the target data (1st: 2D source and target domain features, before transformation, 2nd: after transformation). 3rd and 4th: their distributions.

1 INTRODUCTION

Traditional machine learning (ML) algorithms assume training and test datasets to be *independent and identically distributed* (i.i.d.). Hence, supervised ML only works well when the test data comes from the same distribution as the training data. As real-world data often changes over time and space, this assumption rarely holds in practice [75]. Domain adaptation (DA) [26, 32, 40, 51, 71, 76, 81] as a special instance of transfer learning (TL) [20, 63, 64, 74] tries to compensate for this *domain shift* by transferring knowledge from a *source* to a *target* domain, see Figure 1. There are three types of DA: supervised, semi-supervised, and unsupervised DA. The decision which approach to use mainly depends on the number of labels available in the target domain [91]. Most techniques are unsupervised and transform the source data by minimizing the distance to the target data. Typically, the model is (re-)trained on the transformed source domain [75].

Domain shifts appear in many applications such as classification [25, 34, 45, 68, 86, 90, 96], handwriting recognition [46], segmentation and regression for multimedia data [12, 30], for example if the background, shape deformation, or quality are different across domains. DA aims to mitigate this and has successfully been applied for object recognition [31, 54, 88, 89], AI planning [95], reinforcement learning [56] and natural language processing [93] (e.g., the adaptation from English to Spanish documents [47]).

DA is also used for multivariate time-series (MTS) classification and forecasting [8, 15, 37, 43], which is a challenging task as the extraction of domain-invariant representations is non-trivial.

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

206

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

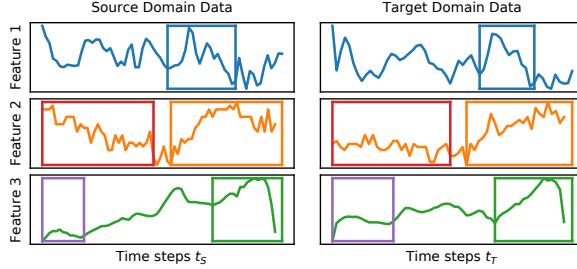


Figure 2: Domain-variant representation of MTS data of a source and target sample from a sensor pen representing the label '5' in an online handwriting task. Note that both samples have different number of time steps t_S (left) and t_T (right). Top row: data from an accelerometer; middle row: gyroscope measures; bottom row: data from a force sensor.

Consider the different time-series of online handwriting from a sensor-enhanced pen in Figure 2 (right: right-handed writers, target domain; left: left-handed writers, source domain) [46, 60, 61]. The shaded areas show the discrepancy between both writers, i.e., different time step lengths and accelerations. In this case, the complex dependency of time steps in the MTS makes it challenging to extract invariant features [8]. Many existing methods [21, 67] employ recurrent neural networks (RNNs) and assume that the conditional distributions of the source and target domain are equal, i.e., $P_S(y|\phi(x_1, \dots, x_{t_S})) = P_T(y|\phi(x_1, \dots, x_{t_T}))$, with the feature transformation mapping $\phi(\cdot)$ and $X = \{x_1, x_2, \dots, x_t\} \in \mathcal{X}$ being a set of training samples with $t \in \{t_S, t_T\}$ time steps [62]. However, this assumption does typically not hold in practice as methods do not generalize across domains without additional efforts.

Classical DA methods range from feature selection [3, 70] (both domains share similarities in the features), distribution adaptation (distributions of both domains are different but share similarities), and subspace learning (a lower-dimensional shared representation). Distribution adaptation methods can be classified into three categories: (1) *Marginal distribution adaptation* methods assume that the marginal distribution between the domains are different and focus on overall shape alignment. While the most established method is maximum mean discrepancy (MMD) [48], e.g., used in [11, 30, 39, 51–53, 81], many further techniques exist [2, 22, 23, 42, 51, 62]. (2) *Conditional distribution adaptation* [73, 85] assumes that the conditional distribution is varied between the domains ($P(\mathcal{Y}_S|\mathcal{X}_S) \neq P(\mathcal{Y}_T|\mathcal{X}_T)$). (3) *Joint distribution adaptation* methods [49, 50] minimize the joint distribution distance between the source and target domain. Subspace alignment methods (i.e., SA [26], CORAL [76], GFK [32]) align the source and target domains via principal component analysis with a lower dimensional space determined by the Bregman divergence. Recently, DL methods have become the predominant approach in DA. Existing techniques are, e.g., based on MMD [48, 81], align the second-order statistics (covariances) [77], use the Kullback-Leibler [94] or Jensen-Shannon divergence [41], or are based on the Wasserstein distance [16, 28, 29]. These methods have been broadly applied to visual object recognition and text categorization [2], but rarely to time-series [8].

We propose a DA method that adapts embeddings from a small source domain (with domain shift) to embeddings from a large target domain (main training dataset on which the model has initially been trained). First, we pre-train our model on a large target domain dataset. Next, we train an optimal transformation T from the source to the target domain for each class with joint distribution adaptation methods, i.e., optimal transport and CORAL, from an adaptation set with few samples. At inference, we extract features with the source domain model, transform features into the target domain for each class, compute the similarity to the target domain to select the best transformation, and classify the transformed embedding with the target domain model. This allows a faster adaptation to new data without the necessity of post-training [35]. We apply this technique to MTS datasets and evaluate the performance of methods for the challenging task of embedding similarity comparison from time-series data [8]. We show performance improvements for synthetically generated univariate sinusoidal data and on multivariate online handwriting (OnHW) datasets from pens with integrated sensors. For the challenging OnHW recognition task of new out-of-distribution writers, we propose a method that can adapt to each writer and outperforms transfer learning approaches.¹

The remainder of this paper is organized as follows. Section 2 discusses related work followed by our proposed methodology in Section 3. The experimental setup is described in Section 4 and the results are discussed in Section 5. Section 6 concludes.

2 RELATED WORK

Research for (multivariate) time-series classification is very advanced and ranges from classical convolutional neural networks (CNNs) such as FCN [86] to advanced CNNs [78, 79] such as ResCNN [96], ResNet [86], XResNet [34], XceptionTime [68] and InceptionTime [25]. Spatio-temporal methods [14, 45, 60] became popular with the development of RNNs, i.e., long short-term memorys (LSTMs) and multi-dimensional LSTMs. Research for Transformers (e.g., TST [90]) for time-series classification is less advanced. As their goal is to classify a (multivariate) time-series without domain shift (e.g., on the UCR [19] datasets), they do not consider to transform embeddings. All such methods have previously been benchmarked on OnHW datasets, and [60] showed a benefit of small CNNs combined with bidirectional LSTMs (BiLSTMs) on OnHW recognition.

Domain Adaptation for Time-Series Classification. Research on MTS-specific DA is rare. The domain adversarial neural network (DANN) learns domain-invariant features [21] and uses a time window approach to extract temporal information from time-series data for prognostics with LSTMs. Similar, the variational recurrent adversarial deep domain adaptation (VRADA) [67] learns domain-invariant temporal relationships based on a variational RNN (VRNN) [15] for MTS healthcare datasets. However, both of them cannot align the condition distribution well. Sparse associative structure alignment (SASA) [8] exploits the sparse associative structure to mitigate the difficult domain-invariant extraction of time-series for offsets (change of time lags, Figure 2). SASA generates a segment set to exclude the obstacle of offsets, extracts

¹Datasets and source code available at: www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html

associative structure time-series data with time lags, and uses the structure alignment for knowledge transfer. The domain adaptation forecaster (DAF) [43] leverages statistical strengths from the source domain to improve the performance on the target domain for forecasting. The attention-based shared module with a domain discriminator across domains induces domain-invariant latent features and retrains domain-specific features. However, all those methods require learning latent features between domains, while applications on embedded devices require approaches to quickly adapt to a new domain at inference. Similar to our approach but for forecasting, Hu et al. [37] use pre-trained cross-domain time-series representations to augment the target domain. Wilson et al. [87] proposed a DA model with weak supervision and a domain classifier with a domain-invariant representation, while we use a target-specific representation.

Covariance- and Discrepancy-based Methods for DA. Procrustes analysis [84] uses the singular value decomposition for manifold alignment, but has a run time complexity of $O(N^3)$ for N samples. Correlation alignment (CORAL) [10, 75, 76] is a simple method and can be used in a (non-)trainable manner for embedding alignment, but has not been used on time-series data before. We make use of CORAL for transformation computations and selection. Many DA methods are based on MMD [48] that minimizes the discrepancy of feature distributions, for example, joint MMD (JMMD) [53] or transfer joint matching (TJM) [51]. Chen et al. [11] proposed a higher-order moment matching (HoMM) technique and extend HoMM into a reproducing kernel Hilbert space (RKHS) [23, 32, 38, 92]. The first order HoMM is equivalent to MMD, and the second-order HoMM is equivalent to CORAL. Higher orders (≥ 3), however, result in long training times. These methods do not directly compute a transformation between source and target domains, but minimize the discrepancy at training time, and hence, we use these methods for transformation selection. Transfer component analysis (TCA) [62] learns transfer components in an RKHS using MMD. Joint distribution adaptation (JDA) [50] reduces the difference between source and target domains in both the marginal and conditional distributions by a principled dimensionality reduction procedure based on MMD. This approach reduces the embedding dimensions and cannot be applied for cross-domain feature classification as given in our application.

Optimal Transport for DA. The earth mover's distance (EMD) and Sinkhorn transport [16, 27, 29] has rarely been used for time-series DA, but successfully for image DA. The Python Optimal Transport (POT) package [28] has been used for adapting domains for the classification of satellite images [80]. Images are considered as time-series as they are an ordered set that is re-sampled onto a regular time grid for consistent length (< 30 images per time-series). For an overview, see [24]. We provide a broad evaluation of EMD and Sinkhorn transport for time-series DA and show benefits over correlation techniques.

Embedding Distances. Related work for comparing embeddings commonly use the Euclidean metric, but also correlation-based metrics are used [58, 59]. Recent methods extend the canonical correlation analysis (CCA) [9, 69] that learn linear projection matrices by maximizing pairwise correlation, but require long computing

times. Metrics typically used for DA can also be utilized as distance metrics between embeddings (e.g., CORAL [75], MMD [11, 48, 53] and HoMM [11]). [58] use the cross- and Pearson correlation [65] for similarity computation. Our evaluation proposes performance comparisons of these methods for the time-series DA application.

3 METHODOLOGY

We start with a formal definition of the notation for domain adaptation (DA) in Section 3.1. We then give an overview of our method that consists of two parts: Optimizing a transformation from source to target domain and selecting the transformation at inference (Section 3.2). An overview of all the introduced notation is additionally given in Appendix A.1.

3.1 Notation

MTS Classification. An MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $l \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l})$, $i \in \{1, \dots, m\}$, where $m \in \mathbb{N}$ is the length of the time-series. The MTS training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\} \in \mathbb{R}^{n_U \times m \times l}$, where n_U is the number of time-series. The aim of MTS classification is to predict an unknown class label $y \in \mathcal{Y}$ for a given MTS. We define the target domain dataset as \mathcal{U}_T . Given a smaller adaptation set of a source domain \mathcal{U}_S with MTS \mathbf{U}_S , the goal of DA is to find an optimal transformation \mathbf{T} of the representation of the latent embedding $f(\mathbf{U}_S)$ of the source domain to the representation of the latent embedding $f(\mathbf{U}_T)$ of the target domain such that the prediction of the unknown class label y_S of the source domain is maximized. $f(\mathbf{U}_T) \in \mathbb{R}^{q_T \times w_T}$ and $f(\mathbf{U}_S) \in \mathbb{R}^{q_S \times w_S}$ are the latent target and source embeddings of the neural network.

Domain Adaptation. A domain \mathcal{D} consists of a feature space \mathcal{X} with marginal probability $P(\mathcal{X})$. The task is defined by the label space \mathcal{Y} . The joint distribution is $P(\mathcal{X}, \mathcal{Y})$ and the conditional distribution is denoted as $P(\mathcal{Y}|\mathcal{X})$. When considering MTS classification, there is a source domain $\mathcal{D}_S = \{\mathcal{X}_S^i, \mathcal{Y}_S^i\}_{i=1}^{N_S}$ of N_S labeled samples of $|\mathcal{Y}_S^i|$ categories, and a target domain $\mathcal{D}_T = \{\mathcal{X}_T^i, \mathcal{Y}_T^i\}_{i=1}^{N_T}$ of N_T labeled samples of $|\mathcal{Y}_T^i|$ categories. Due to the difference of the two domains, the distributions are assumed to be different: $P(\mathcal{X}_S) \neq P(\mathcal{X}_T)$ and $P(\mathcal{Y}_S|\mathcal{X}_S) \neq P(\mathcal{Y}_T|\mathcal{X}_T)$ (see Figure 1, left) [91]. DA can mitigate the domain shift and improve the classification accuracy in the target domain (see Figure 1, 2nd and 4th).

3.2 Method Overview

An overview of our method for MTS classification is given in Figure 3 and in Algorithm 1. We train a convolutional neural network with two BiLSTM layers on the target data, and train a different (but same) model on the (training) source data. Next, we search for the optimal transformation \mathbf{T} to transform the feature embeddings of the source data onto the feature embeddings of the target data. As feature embeddings $f(\mathbf{U}_S)$ and $f(\mathbf{U}_T)$ we choose the output of the last convolutional layer before the two BiLSTMs of size $\mathbb{R}^{50 \times 30}$ and $\mathbb{R}^{19 \times 200}$. We obtain a separate transformation for each class. To find the optimal transformation, we evaluate different DA techniques, see Section 3.2.1. At inference time, we extract features of the (validation) source data with the source domain model, transform the embedding with each class-dependent transformation, and choose

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

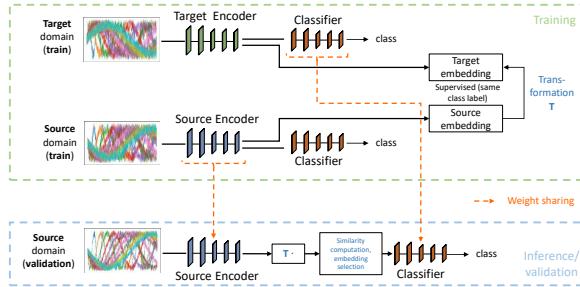


Figure 3: Method overview. We transform the feature embeddings of the source domain (training data, few samples) into feature embeddings of the target domain (training adaptation data, many samples) with optimal transport and correlation alignment. For inference, we evaluate the feature similarity.

Algorithm 1 Domain adaptation: Source data transformation and transformation selection.

Input: Target data \mathcal{U}_{T_t} , source data \mathcal{U}_{S_t} and \mathcal{U}_{S_v} , and labels \mathcal{Y}_{T_t} , \mathcal{Y}_{S_t} and \mathcal{Y}_{S_v}
Output: Class predictions for source validation data \mathcal{U}_{S_v}

```

1: function DOMAINADAPTATION( $\mathcal{U}_{T_t}$ ,  $\mathcal{U}_{S_t}$ ,  $\mathcal{U}_{S_v}$ )
2:   Train target model with data  $\mathcal{U}_{T_t}$  and labels  $\mathcal{Y}_{T_t}$ 
3:   Train source model with data  $\mathcal{U}_{S_t}$  and labels  $\mathcal{Y}_{S_t}$ 
4:   for  $i \leftarrow 1$  to  $|\mathcal{Y}_{S_t}|$  do
5:     Compute transformation  $T_i$  between  $\mathcal{U}_{T_t,i}$  and  $\mathcal{U}_{S_t,i}$ 
6:     ▷ See Section 3.2.1
7:   end for
8:   for  $i \leftarrow 1$  to  $|\mathcal{U}_{S_t}|$  do
9:     for  $k \leftarrow 1$  to  $|\mathcal{Y}_{S_v}|$  do
10:    Apply transformation  $T_k$  to source data  $\mathcal{U}_{S_v,i}$ 
11:    Compute similarity between  $\mathcal{U}_{S_v,i}(T_k)$  and  $\mathcal{U}_{T_t,i}$ 
12:    ▷ See Section 3.2.2
13:   end for
14:   Select  $T$  with highest similarity of  $\mathcal{U}_{S_v,i}(T)$  and  $\mathcal{U}_{T_t,i}$ 
15:   Predict class label  $y_{S_v}$  of transformed  $\mathcal{U}_{S_v,i}(T)$ 
16: end for
17: end function

```

the transformation for the closest embedding to the target domain data. For source and target domain similarity computation, see Section 3.2.2. Lastly, we classify the transformed embedding with the target domain classifier with the cross-entropy (CE) loss.

3.2.1 Embedding Transformation. The goal of DA is to minimize the target domain error by bounding the source domain error and the discrepancy between them [4–6]. DA approaches consider the target data for optimizing the source domain model and reduce the discrepancy between them as in Theorem 1:

THEOREM 1 (ZHANG, 2021 [91]). Let \mathcal{H} be a hypothesis space. Given the target domain \mathcal{D}_T and source domain \mathcal{D}_S , we have

$$\forall h \in \mathcal{H}, \mathcal{R}_T(h) \leq \mathcal{R}_S(h) + d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \beta, \quad (1)$$

where $\mathcal{R}_T(h)$ is the target domain error and $\mathcal{R}_S(h)$ is the source domain error. $d_{\mathcal{H}\Delta\mathcal{H}}$ is the discrepancy distance between \mathcal{D}_S and \mathcal{D}_T w.r.t. \mathcal{H} . Then, given the label functions g_T and g_S determined by the domain labels \mathcal{Y}_T and \mathcal{Y}_S , the shared error β is

$$\beta = \arg \min_{h \in \mathcal{H}} \mathcal{R}_S(h^*, g_S) + \mathcal{R}_T(h^*, g_T), \quad (2)$$

where h^* is the ideal hypothesis.

Homogeneous Domain Adaptation. DA models aim to find a minimal discrepancy distance $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T)$. For classifying time-series data, we restrict our methods to *homogeneous* DA, where the feature space is the same ($\mathcal{U}_T = \mathcal{U}_S$) with the same feature dimensionality ($q_T = q_S$ and $w_T = w_S$) and interpolate the time-series to a pre-defined fixed length. In the following, we provide details on computing the optimal transformation.

Optimal Transport. To mitigate the domain shift for our multivariate time-series application, we assume that the domain drift is due to an unknown, possibly nonlinear map of the input space $T : \mathcal{D}_S \rightarrow \mathcal{D}_T$ that preserves the conditional distribution $P_S(y|f(\mathcal{U}_S)) = P_T(y|f(\mathcal{U}_S))$ such that the label information is preserved [16]. Searching for T in the space of all possible transformations is intractable. Hence, T is chosen such that a transportation cost

$$C(T) = \int_{\mathcal{D}_S} c(f(\mathcal{U}), T(f(\mathcal{U}))) d\mu(f(\mathcal{U})), \quad (3)$$

is minimized, where $c : \mathcal{D}_T \times \mathcal{D}_S \rightarrow \mathbb{R}^+$ is a distance function over the metric space \mathcal{D} [16]. The optimal transportation problem is

$$T_0 = \arg \min_T \int_{\mathcal{D}_S} c(f(\mathcal{U}), T(f(\mathcal{U}))) d\mu(f(\mathcal{U})). \quad (4)$$

This is also known as the Kantorovich formulation [44] that allows to search a general coupling $\alpha \in \Theta$ by the *transportation plan* [72]:

$$\alpha_0 = \arg \min_{\alpha \in \Theta} \int_{\mathcal{D}_T \times \mathcal{D}_S} c(f(\mathcal{U}_T), f(\mathcal{U}_S)) d\alpha(f(\mathcal{U}_T), f(\mathcal{U}_S)), \quad (5)$$

where Θ is a set of all probabilistic couplings $\Theta \in P(\mathcal{D}_T \times \mathcal{D}_S)$ with marginals μ_T and μ_S . Then, the *Wasserstein distance* of order p between μ_T and μ_S can be defined as

$$W_p(\mu_T, \mu_S) := \left(\inf_{\alpha \in \Theta} \int_{\mathcal{D}_T \times \mathcal{D}_S} d(f(\mathcal{U}_T), f(\mathcal{U}_S))^p d\alpha(f(\mathcal{U}_T), f(\mathcal{U}_S)) \right)^{\frac{1}{p}}, \quad (6)$$

where d is a distance metric [16] as the cost function:

$$c(f(\mathcal{U}_T), f(\mathcal{U}_S)) = d(f(\mathcal{U}_T), f(\mathcal{U}_S))^p. \quad (7)$$

We apply the earth mover's distance (EMD), the EMD with Laplacian regularization [29], and Sinkhorn transport [1, 17] (with $L_p L_1$ and $L_1 L_2$ class regularization of 0.5) implemented by the Python Optimal Transport (POT) package [28] between our source and target domain samples. Selecting a proper cost function is crucial for the effectiveness of the adaptation of source to target domain. Typically, the L_2 -based metric is used, but other cost functions are also possible, e.g., norm-based metrics [83], metrics based on Riemannian distances over a manifold [83], metrics used as a loss function [18], or concave cost functions [27]. We evaluate 18 different distance metrics in Section 5.3. For comparison, we use correlation alignment [75] (for more information, see Appendix A.2).

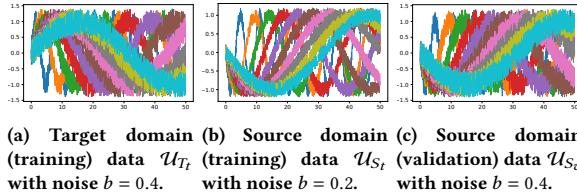


Figure 4: Time-series of the generated sinusoidal dataset.

3.2.2 Transformation Selection for Inference. At inference we aim to select the best class-specific transformation (while the class label of the sample is not known). Hence, we compute the transformation of the validation source domain embeddings for each class, and select the transformation for which the embeddings of the target $f(U_T)$ and transformed source $T(f(U_S))$ domains have the smallest cost $c(T(f(U_S)), f(U_T))$. We use common similarity metrics such as the cross correlation (CC) d_{CC} and Pearson correlation (PC) [65] d_{PC} . We also compute the MMD [48] by

$$d_{MMD}(\mathcal{D}_T^C, \mathcal{D}_S^C) = \left\| \frac{1}{N_T} \sum_{i=1}^{N_T} f(U_T^{C,i}) - \frac{1}{N_S} \sum_{j=1}^{N_S} T(f(U_S^{C,j})) \right\|_{\mathcal{H}}^2, \quad (8)$$

for class C and the RKHS \mathcal{H} . We compare to HoMM [11] of order 3 and kMMD [53] (which is equivalent to the kernelized HoMM of order 1). We also make use of different CORAL metrics (standard, Stein [13] and Jeff [55] CORAL based on symmetrized Bregman divergences [33]) for embedding comparisons.

4 EXPERIMENTAL SETUP

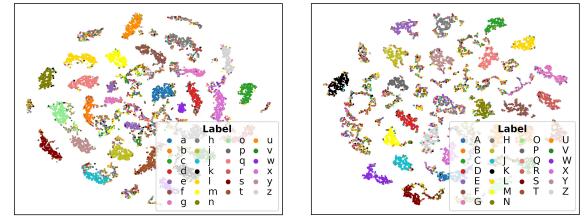
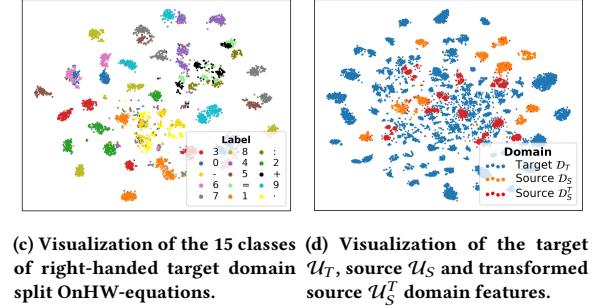
We apply our DA technique to two time-series datasets: generated time-series (Section 4.1), and OnHW recognition (Section 4.2).

4.1 Synthetic Time-Series Classification

We first investigate the efficacy of our method to transform synthetically generated univariate time-series data. For this, we generate sinusoidal signal data of 200 time steps with different frequencies for 10 classes. We add noise from a continuous uniform distribution $U(a, b)$ with $a = 0.0$ and $b \in B = \{0.0, 0.1, 0.2, \dots, 1.9\}$ for the target domain dataset (Figure 4a), and train a CNN+BiLSTM. For the source domain (training) dataset, we flip the sign of the generated time-series and add uniform $U(a, b/2)$ noise (Figure 4b). We then validate the adapted model using the flipped dataset with added uniform noise for $b \in B$ values (Figure 4c). This allows us to evaluate the time-series adaptation for different noise ratios.

4.2 Online Handwriting (OnHW) Recognition

OnHW recognition typically uses time in association with different types of spatio-temporal signals. The data contains information about the displacement of certain input devices [66]. OnHW recognition from sensor-enhanced pens uses data from inertial measurement units to capture the pen movement. The pen in [61] uses two accelerometers, one gyroscope, one magnetometer, 3 axes each, and one force sensor at 100 Hz. One sample of size $m \times l$ represents an MTS of m time steps from $l = 13$ sensor channels. We make use of three character-based datasets: The *OnHW-chars* [61] dataset

Figure 5: Embedding $f(U)$ visualization of OnHW-chars (a and b) and split OnHW-equations (c and d) using t-SNE.

contains 31,275 samples of small and capital characters (52 label classes) from 119 right-handed writers. The *OnHW-symbols* [60] dataset contains 2,326 samples of numbers and symbols (15 label classes) from 27 right-handed writers, and the *split OnHW-equations* [60] dataset contains 39,643 numbers and symbols from 55 right-handed writers obtained from 10,713 equations. For these datasets, 80/20 train/validation splits are available for writer-dependent (WD) and writer-independent (WI) classification tasks. We define these datasets as our target domain U_T . Usually, observations from left-handed writers in OnHW datasets is scarce [46]. As we want OnHW recognition to work equally well for left-handed writers as for right-handed writers, we use the smaller available left-handed datasets as source domain U_S and split it into an adaptation (training) set U_{S_t} and a validation set U_{S_v} . For an overview, see Appendix A.3.

Figure 5 visualizes the 19×200 dimensional feature embeddings of the CNN+BiLSTM model for the OnHW-chars and split OnHW-equations datasets. We use the t-SNE method [82] with an initial dimension 3,800, perplexity of 30, an initial momentum of 0.5, and a final momentum of 0.8. Figure 5a and 5b visualizes 26 lower and 26 uppercase class labels of OnHW-chars. Several samples are in clusters associated with a different class and hence wrongly classified. Figure 5c differentiates between all 15 class labels for the split OnHW-equations dataset. Here, we can clearly see that the labels '+', '-' and ':' are close in their low-dimensional embedding. Figure 5d shows the low-dimensional embedding of the right-handed target domain D_T (blue) and the left-handed source domain D_S (orange). It is notable that the features of both domains have different distributions. After the transformation, the left-handed source domain D_S^T (red) is closer to the target domain.

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift 210

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

Table 1: Evaluation results (CRR in %, mean and standard deviation) for transfer learning techniques on the OnHW-symbols and split OnHW-equations datasets [60] (averaged over four left-handed writers) and on the OnHW-chars [61] dataset (averaged over nine left-handed writers) based on the CNN+BiLSTM architecture.

Method	OnHW-symbols [60]	split OnHW-equations [60]	OnHW-chars [61]		
			lower	upper	combined
Baseline (\mathcal{U}_{S_t} in Target Model)	19.18	33.52	45.80	45.97	25.19
Without Transformation	36.92 ± 8.71	47.06 ± 27.13	3.60 ± 2.35	3.87 ± 5.80	3.03 ± 2.44
Post-Training (full)	27.49 ± 14.29	90.53 ± 6.12	35.85 ± 30.33	31.15 ± 30.20	50.22 ± 19.74
Post-Training (middle)	19.23 ± 16.00	88.00 ± 6.51	28.39 ± 23.54	21.42 ± 23.92	41.76 ± 18.81
Post-Training (last)	22.15 ± 20.47	78.87 ± 10.89	16.17 ± 14.31	10.75 ± 14.99	19.88 ± 14.20
Layer Adapting	6.87 ± 2.45	21.53 ± 4.60	2.67 ± 2.04	4.87 ± 2.87	2.10 ± 2.56

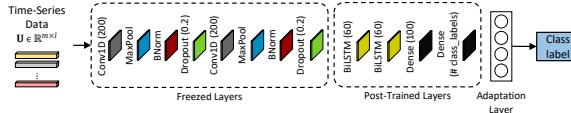


Figure 6: Network architecture with transfer learning by freezing the first layers and fine-tuning the last layers, or only post-training an additional adaptation layer. The network is pre-trained on the target domain \mathcal{U}_{T_t} , and adapted on the source domain training set \mathcal{U}_{S_t} and validated on the set \mathcal{U}_{S_v} .

5 EXPERIMENTAL RESULTS

Hardware and Training Setup. For all experiments we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the vanilla Adam optimizer with a learning rate of 10^{-4} . We use the character recognition rate (CRR) in % as time-series classification evaluation metric. Details on the architecture, are proposed in Appendix A.7.

5.1 Limitations of Transfer Learning (TL)

Fine-tuning an existing model on each source domain can be data- and parameter-inefficient. Under the assumption that the adaptation dataset is large enough, fine-tuning leads to a better performance than adapting the domains by feature-based transfer as the model can overfit on the source data [36]. Hence, we apply different TL techniques and compare to DA techniques. Figure 6 shows our network architecture and different TL techniques. First, we adapt the whole model by fine-tuning on each source domain dataset from the pre-trained network on the target domain dataset. Second, we freeze the first layers and only post-train the spatio-temporal layers (two BiLSTMs and two dense layers). This layer freezing leads to a faster training than full post-training. Third, we apply a structurally similar method to the Adapter by Houlsby et al. [35]: We freeze all previously pre-trained layers, add an adaptation layer (a standard *dense* layer) at the end, and only train the additional layer. The *dense* layer has $|\mathcal{Y}|$ units. This yields a compact and fast trainable model by adding only a few trainable parameters per task.

Sinusoidal Dataset Evaluation. We adapt the pre-trained models (on the source domain datasets) with the target domain datasets by post-training the models provided in Figure 6. Figure 7 shows the results averaged over 10 trainings for all noise parameters $b \in B$.

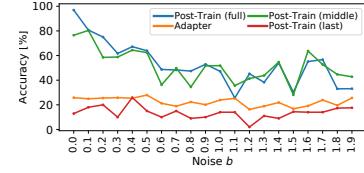


Figure 7: Evaluation of transfer learning on the generated dataset for noise parameters between $b = 0.0$ and $b = 1.9$.

Post-training the full model and freezing the first layers while post-training the last layers yields the highest TL results. Post-training only the last layer or adapting an additional layer results in low classification accuracies below 20%. In general, TL performs poorly when we see significant changes between the domains (i.e., $b > 0.5$). In particular, all results are lower than the results achieved by our DA techniques (see Section 5.2, Figure 8).

OnHW Recognition Evaluation. Table 1 shows TL results for all OnHW datasets. Without transformation, the models fail in the classification tasks, which proves the existence of a domain shift in the data [46]. As the OnHW-symbols dataset is rather small, all TL techniques cannot adapt to a specific writer. The larger OnHW-chars dataset leads to better results. TL only yields good results on the split OnHW-equations dataset, while full model post-training outperforms training only specific layers or adapting an additional layer. Again, TL is limited in its efficacy while our DA approach shows promising results (see Section 5.3).

5.2 Evaluation of Sinusoidal Data

We train each sinusoidal dataset 10 times, and present results of mean and standard deviation. As a gold standard we apply the transformation based on the known label class, which we define as the *upper bound* for transformation selection, and define the *lower bound* by the classification without the use of any transformation.

Figure 8 presents results for the five optimal transport techniques. The classification accuracy notably drops for higher noise rates ($b > 0.7$). Without transformation (*lower bound*) the accuracy is below 20% (see Figure 8a, cyan). The *upper bound* (black line) yields an accuracy between 80% and 100%. MMD is outperformed by all transformation selection techniques as order 1 is not suitable. For the remaining techniques, the Laplacian regularization improves EMD results, while regularizing Sinkhorn ($L_p L_1$ and $L_1 L_2$)

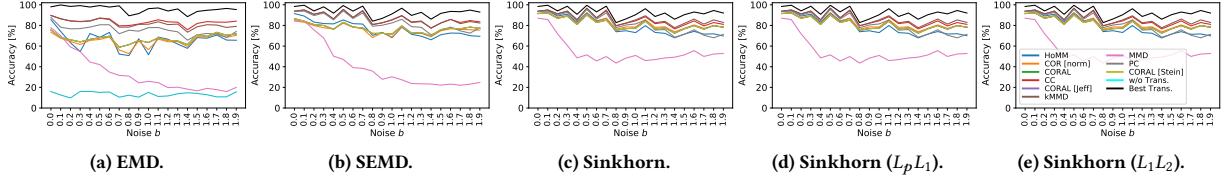


Figure 8: Results for optimal transport methods and transformation selection on the synthetic dataset (for b between 0.0 and 1.9). For better readability we depict the approach without transformation only in (a) as it is independent of optimal transport.

Table 2: Evaluation results (CRR in %) for the left- and right-handed writer OnHW-symbols and split OnHW-equations datasets [60] for different MTS classification techniques. \mathcal{U}_{T_v} are right-handed writer datasets, and \mathcal{U}_{S_v} are left-handed writer datasets.

Method	OnHW-symbols [60]						split OnHW-equations [60]					
	Right-handed		Left-handed		L in R		Right-handed		Left-handed		L in R	
	WD	WI	WD	WI	WI	WD	WD	WI	WD	WI	WI	
CNN+BiLSTM [60]	96.20	79.51	92.00	54.00	19.18	95.70	83.88	92.00	51.50	33.52		
LSTM-FCN [45]	92.39	73.32	75.34	41.40	-	93.95	81.47	88.56	47.56	-		
ResCNN ($nf = 64$) [96]	92.23	77.41	80.82	47.87	-	94.58	80.95	89.59	40.45	-		
ResNet ($nf = 64$) [86]	94.50	77.41	80.82	47.87	-	94.68	83.45	89.20	39.21	-		
XResNet50 [34]	93.66	74.47	78.08	47.87	-	94.63	81.74	89.67	45.15	-		
XceptionTime ($nf = 16$) [68]	91.54	72.34	75.34	40.43	-	94.03	82.24	88.72	50.73	-		
InceptionTime ($nf = 64$, depth = 12) [25]	91.97	76.92	80.82	46.81	-	94.87	84.35	88.48	44.15	-		
TST [90]	91.12	71.85	78.08	51.06	-	93.07	80.40	87.61	47.27	-		

Table 3: Evaluation results (CRR in %) for the left-handed (\mathcal{U}_{S_v}) and right-handed (\mathcal{U}_{T_v}) writer OnHW-chars [61] datasets for different time-series classification techniques with same parameters as in Table 2.

Method	Right-handed (\mathcal{U}_{T_v})						Left-handed (\mathcal{U}_{S_v})						\mathcal{U}_{S_v} in Target Model		
	Lower		Upper		Combined		Lower		Upper		Combined		Lower	Upper	Combined
	WD	WI	WD	WI	WD	WI	WD	WI	WD	WI	WD	WI	WI	WI	WI
CNN+BiLSTM [60]	88.85	79.48	92.15	85.60	78.17	68.06	94.70	43.60	91.90	43.62	82.80	32.00	45.80	45.97	25.19
LSTM-FCN [45]	81.43	71.41	85.43	77.07	67.34	57.93	70.55	34.06	72.50	29.27	61.02	22.68	-	-	-
ResCNN ($nf = 64$) [96]	82.52	72.00	86.91	78.64	67.55	58.67	80.00	38.78	80.63	29.79	65.39	26.21	-	-	-
ResNet ($nf = 64$) [86]	83.01	71.93	86.41	78.03	68.56	58.74	81.01	40.24	82.95	30.12	66.95	26.17	-	-	-
XResNet50 [34]	80.99	69.14	86.05	76.69	64.98	54.38	74.86	31.24	76.43	28.35	60.80	18.38	-	-	-
XceptionTime ($nf = 16$) [68]	81.41	70.76	85.94	78.23	66.70	56.92	75.41	40.08	79.20	30.66	63.92	25.91	-	-	-
InceptionTime ($64, 12$) [25]	84.14	75.28	87.80	81.62	70.43	61.68	79.08	43.12	81.25	36.48	65.12	29.35	-	-	-
TST [90]	80.10	70.75	84.81	78.34	66.12	57.56	77.43	41.27	79.11	29.86	63.39	26.83	-	-	-

does not yield better results compared to standard Sinkhorn. Consistently, CC and kMMD outperform CORAL which yields higher accuracies than HoMM of order three and Pearson correlation. For the evaluation of feature embeddings, we refer to Appendix A.4.

5.3 Evaluation of OnHW Recognition

Baseline Results. We train all three OnHW right- and left-handed writer datasets for writer-dependent (WD) and writer-independent (WI) tasks. Architectures are taken from [60] (CNN+ BiLSTM) and the tsai toolbox [57] (for all other models) [25, 34, 45, 68, 86, 90, 96]. Results of these comparisons are given in Tables 2 and 3. For the OnHW-symbols dataset, the CNN+BiLSTM model outperforms all architectures, while for the split OnHW-equations dataset, InceptionTime [25] outperforms the CNN+BiLSTM architecture on the right-handed WI task. On the OnHW-chars dataset, the accuracy of the CNN+BiLSTM model is notably higher compared to all other models, and we hence choose the CNN+BiLSTM model for further

experiments. Simply classifying the left-handed writer samples with the model pre-trained on right-handed writer data yields low accuracies (19.18% on the OnHW-symbols dataset and 33.52% on the split OnHW-equations dataset) as left-handed writer samples are out-of-distribution with respect to the right-handed ones (see Figure 5d). Hence, an efficient DA technique is necessary.

Evaluation of Domain Adaptation. We first evaluate different distance metrics for optimal transport. The respective hyperparameter search results are given in Appendix A.5. Based on these findings, we choose the squared Euclidean metric for all further applications. Table 4 summarizes all results for DA pre-trained on the target domain data \mathcal{U}_{T_v} , adapted on the source domain data \mathcal{U}_{S_t} , and validated on \mathcal{U}_{S_v} . We train each left-handed writer separately and report average results with their respective standard deviation. For an evaluation for each left-handed writer, see Appendix A.6. The last column in Table 4 shows the *upper bound* using the known transformation. EMD [27] and SEMD [29] perform similar,

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

Table 4: Evaluation results (CRR in %, mean and standard deviation) for different transformation techniques and transformation selections for all OnHW [60, 61] datasets based on the CNN+BiLSTM architecture. J = Jeff, S = Stein.

Dataset	Method	CC [65]	PC [65]	MMD [48]	kMMD [53]	HoMM [11]	CORAL	CORAL (J)	CORAL (S)	w/ T.
OnHW-symbols	EMD [27]	70.03± 8.4	59.14±13.4	61.55±18.2	85.02±6.9	67.11±6.8	75.29±16.0	82.24±7.1	80.85±8.3	89.18±10.8
	SEMD [29]	70.03± 8.4	59.14±13.4	61.55±18.2	85.02±6.9	65.72±8.3	75.29±16.0	82.24±7.2	80.85±8.4	89.18±10.8
	Sinkhorn [16]	64.62±19.2	51.97± 5.2	67.25± 8.1	85.09±7.7	70.03±7.4	78.14± 6.5	80.92±8.0	82.31±7.8	93.35± 8.6
	Sink. ($L_p L_1$)	64.62±19.2	51.97± 5.2	67.25± 8.1	85.09±7.7	69.96±5.6	78.14± 6.5	80.92±8.0	82.31±7.8	93.35± 8.6
	Sink. ($L_1 L_2$)	64.62±19.2	51.97± 5.2	67.25± 8.1	85.09±7.7	68.64±7.3	78.14± 6.5	80.92±8.0	82.31±7.8	93.35± 8.6
	CORAL [75]	5.48± 6.8	2.78± 2.8	5.49± 0.1	6.80±2.2	5.49±0.1	9.66± 7.3	5.49±0.1	4.17±2.4	39.77±10.0
split OnHW-equations	EMD [27]	68.90±14.4	56.16±20.3	59.33±14.9	79.00±11.7	53.87±10.5	73.61±11.4	77.39±11.6	72.09±13.2	85.78± 7.7
	SEMD [29]	68.90±14.4	56.16±20.3	59.33±14.9	79.00±11.7	53.05±10.0	73.61±11.4	77.39±11.6	72.09±13.2	85.78± 7.7
	Sinkhorn [16]	69.69±17.1	53.31±16.8	67.12± 9.9	84.03± 9.4	62.37±13.3	79.97±10.8	82.24±10.3	76.12±13.0	90.26± 6.3
	Sink. ($L_p L_1$)	69.69±17.1	53.31±16.8	67.12± 9.9	84.03± 9.4	61.86±13.2	79.97±10.8	82.24±10.3	76.12±13.0	90.26± 6.3
	Sink. ($L_1 L_2$)	69.69±17.1	53.33±16.8	67.12± 9.9	84.03± 9.4	62.29±13.4	79.97±10.8	82.24±10.3	76.12±13.0	90.26± 6.3
	CORAL [75]	22.19±12.2	15.82±12.6	15.05± 8.4	19.63±13.0	19.01± 9.0	18.27±10.8	18.18±11.2	17.72±11.8	30.34±11.0
OnHW-chars (lower)	EMD [27]	68.41±13.7	54.27± 8.6	33.06±13.7	77.72±11.0	50.51± 9.5	64.25±10.0	80.08±9.5	79.41±9.5	98.19±2.1
	SEMD [29]	68.41±13.7	54.27± 8.6	33.06±13.7	77.72±11.0	50.57± 9.5	64.25±10.0	80.08±9.5	79.41±9.5	98.19±2.1
	Sinkhorn [16]	81.56±12.3	69.27±12.6	51.54±16.6	83.55±10.1	67.31±10.5	77.36±11.3	82.33±9.5	85.22±8.2	100.00±0.0
	Sink. ($L_p L_1$)	81.56±12.3	69.27±12.6	51.54±16.6	83.55±10.1	66.88±11.1	77.36±11.3	82.33±9.5	85.22±8.2	100.00±0.0
	Sink. ($L_1 L_2$)	81.56±12.3	69.27±12.6	51.54±16.6	83.55±10.1	67.96±10.0	77.36±11.3	82.32±9.5	85.22±8.2	100.00±0.0
	CORAL [75]	4.56± 4.1	8.65± 6.3	4.28± 0.4	3.87± 3.9	4.28± 0.4	4.28± 0.4	4.28± 0.4	7.40±4.3	47.13±9.2
OnHW-chars (upper)	EMD [27]	62.98±18.1	53.37±15.1	33.93±10.2	67.94±13.1	44.06±12.0	60.70±14.0	79.87±15.3	76.29±16.0	99.31± 2.0
	SEMD [29]	62.98±18.1	53.37±15.1	33.93±10.2	67.94±13.1	42.67±12.3	60.70±14.0	79.87±15.3	76.29±16.0	99.31± 2.0
	Sinkhorn [16]	78.82±13.0	73.32±15.5	55.82±17.6	82.25±14.5	57.57±11.0	74.56±12.6	80.96±16.3	78.25±14.2	99.65± 1.0
	Sink. ($L_p L_1$)	78.82±13.0	73.32±15.5	55.82±17.6	82.26±14.5	57.57±11.3	74.56±12.6	80.96±16.3	78.25±14.2	99.65± 1.0
	Sink. ($L_1 L_2$)	78.82±13.0	73.32±15.5	55.82±17.6	82.26±14.5	57.34±10.2	74.56±12.6	80.96±16.3	78.25±14.2	99.65± 1.0
	CORAL [75]	7.03± 4.6	21.51±10.7	4.26± 1.7	6.27± 3.3	4.26± 1.7	4.26± 1.0	4.26± 1.0	11.11± 6.4	52.10±16.5
OnHW-chars (comb.)	EMD [27]	62.18±12.1	51.79±12.9	31.57±11.7	69.62± 9.3	45.21±11.6	57.79±11.4	69.56±9.0	63.96±10.4	94.97±3.4
	SEMD [29]	62.18±12.1	51.79±12.9	31.57±11.7	69.62± 9.3	42.80±12.2	57.79±11.4	69.56±9.1	63.96±10.4	94.98±3.4
	Sinkhorn [16]	65.16±12.1	62.92±10.4	44.16±14.7	73.87±14.7	56.84± 9.6	67.85±10.5	72.30±9.1	66.50± 7.7	96.25±3.5
	Sink. ($L_p L_1$)	65.16±12.1	62.93±10.4	44.16±14.7	73.87±11.1	55.90±10.1	67.85±10.5	72.30±9.1	66.50± 7.7	96.25±3.5
	Sink. ($L_1 L_2$)	65.16±12.1	62.92±10.4	44.16±14.7	73.87±11.1	56.48± 9.2	67.85±10.5	72.30±9.1	66.50± 7.7	96.25±3.5
	CORAL [75]	3.35± 1.2	4.11± 2.6	0.87± 1.3	3.03± 1.8	0.87± 1.3	0.87± 1.0	0.87±1.3	3.24± 2.2	32.41±8.7

as well as Sinkhorn [16] without and with ($L_p L_1, L_1 L_2$) regularization. Sinkhorn transport consistently outperforms EMD (see also [1]). The model with CORAL [75] for transformation computation fails to classify the time-series data, even with known transformation selection. The MMD [48] approach (of order 1) yields the lowest classification accuracy, and is notably be improved with CORAL [75] (of order 2). Increasing the order to 3 (HoMM [11]), decreases the accuracy as a higher number of iterations is required. The kernalized MMD (kMMD) [53] approach (of order 1) yields the highest classification accuracies. Kernelizing HoMM leads to extremely long runtimes. Jeff and Stein CORAL outperform the standard CORAL method as these are not dependent on its inverse [33]. Interestingly, CC performs better than the scale-invariant PC.

Runtimes. We demonstrate runtimes for transformation computation and selection methods exemplary on the OnHW-symbols dataset for one writer averaged over all samples. To find the optimal transformation, EMD (0.0033s) and Sinkhorn (0.0042s) are the fastest methods, while the regularization increases the computation time (SEMD: 1.8756s, Sinkhorn $L_p L_1$: 0.025, $L_1 L_2$: 0.0526s). CORAL leads to extremely long runtimes of 405.7s. Applying the transformation gives the following runtimes in ascending order: CC (0.013s), MMD (0.013s), PC (0.021s), kMMD (0.024s), CORAL (0.03s), CORAL (S) (0.109s), CORAL (J) (0.181s), HoMM (2.844s). We

conclude that Sinkhorn with kMMD is the best trade-off between classification accuracy and runtime.

6 CONCLUSION

We addressed DA for time-series classification by combining a large variety of DA techniques with transformation selection methods. We used optimal transport and correlation alignment techniques to transform features of a source domain into features of a target domain. At inference, we compared correlation metrics and methods based on MMD as embedding distance metrics to select the optimal transformation. A broad study on synthetic univariate time-series data and MTS OnHW datasets showed that Sinkhorn transport can outperform EMD and CORAL. The kernalized MMD metric yields the highest classification accuracies. Our DA approach yields higher accuracies than transfer learning on small adaptation datasets.

ACKNOWLEDGMENTS

Supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A (David Rügamer) and by the research program Human-Computer-Interaction through the project “Schreibtrainer”, Grant No. 16SV8228, as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

REFERENCES

[1] Jason Altschuler, Jonathan Weed, and Philippe Rigollet. 2017. Near-Linear Time Approximation Algorithms for Optimal Transport via Sinkhorn Iterations. In *Advances in Neural Information Processing Systems (NIPS)*. 1961–1971.

[2] Mahsa Baktashmotlagh, Mehrtash Harandi, and Mathieu Salzmann. 2016. Distribution-Matching Embedding for Visual Domain Adaptation. In *Journal of Machine Learning Research (JMLR)*, Vol. 17. 1–30.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. 2006. SURF: Speeded Up Robust Features. In *Euro. Conf. on Computer Vision (ECCV)*, Vol. 3951. 404–417. https://doi.org/10.1007/11744023_32

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortmann Vaughan. 2009. A Theory of Learning from Different Domains. In *Machine Learning*, Vol. 79. 151–175.

[5] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of Representations for Domain Adaptation. In *Advances in Neural Information Processing Systems (NIPS)*.

[6] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortmann. 2007. Learning Bounds for Domain Adaptation. In *Advances in Neural Information Processing Systems (NIPS)*.

[7] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. 2008. A Singular Value Thresholding Algorithm for Matrix Completion. In [arXiv:0810.3286](https://arxiv.org/abs/0810.3286).

[8] Ruichu Cai, Jiawei Chen, Zijian Li, Wei Chen, Keli Zhang, Junjian Ye, Zhuozhang Li, Xiaoyan Yang, and Zhenjie Zhang. 2014. Time Series Domain Adaptation via Sparse Associative Structure Alignment. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 216. 76–102.

[9] James Chapman and Hao-Ting Wang. 2021. CCA-Zoo: A Collection of Regularized, Deep Learning based, Kernel, and Probabilistic CCA Methods in a scikit-learn Style Framework. In *Journal of Open Source Software (JOSS)*, Vol. 6(68).

[10] Chao Chen, Zhihong Chen, Boyuan Jiang, and Xinyu Jin. 2019. Joint Domain Alignment and Discriminative Feature Learning for Unsupervised Deep Domain Adaptation. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 33(1). 3296–3303.

[11] Chao Chen, Zhihong Fu, Zhihong Chen, Sheng Jin, Zhaowei Cheng, Xinyu Jin, and Xian sheng Hua. 2020. HoMM: Higher-Order Moment Matching for Unsupervised Domain Adaptation. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 34(4). 3422–3429.

[12] Long Chen, Hanwang Zhang, Jun Xiao, Wei Liu, and Shih-Fu Chang. 2018. Zero-Shot Visual Recognition Using Semantics-Preserving Adversarial Embedding Networks. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT, 1043–1052. <https://doi.org/10.1109/CVPR.2018.00115>

[13] Anoop Cherian, Suvrit Sra, Arindam Banerjee, and Nikolaos Papanikolopoulos. 2012. Jensen-Bregman LogDet Divergence with Application to Efficient Similarity Search for Covariance Matrices. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 35(9). 2161–2174.

[14] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In [arXiv:1412.3555](https://arxiv.org/abs/1412.3555).

[15] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron Courville, and Yoshua Bengio. 2015. A Recurrent Latent Variable Model for Sequential Data. In *Advances in Neural Information Processing Systems (NIPS)*.

[16] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. 2016. Optimal Transport for Domain Adaptation. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 39(9). 1853–1865. <https://doi.org/10.1109/TPAMI.2016.2615921>

[17] Marco Cuturi. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *Advances in Neural Information Processing Systems (NIPS)*.

[18] Marco Cuturi and David Avis. 2014. Ground Metric Learning. In *Journal of Machine Learning Research (JMLR)*, Vol. 15. 533–564.

[19] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. 2018. The UCR Time-Series Archive. In [arXiv:1810.07758](https://arxiv.org/abs/1810.07758).

[20] Oscar Day and Taghi M. Khoshgoftaar. 2017. A Survey on Heterogeneous Transfer Learning. In *Journal of Big Data*, Vol. 4(29).

[21] Paulo Roberto de Oliveira da Costa, Alp Akcay, Yingqian Zhang, and Uzay Kaymak. 2020. Remaining Useful Lifetime Prediction via Deep Domain Adaptation. In *Reliability Engineering & System Safety*, Vol. 195. <https://doi.org/10.1016/j.ress.2019.106682>

[22] Fatemeh Dorri and Ali Ghodsi. 2012. Adapting Component Analysis. In *Intl. Conf. on Data Mining*. Brussels, Belgium, 846–851. <https://doi.org/10.1109/ICDM.2012.85>

[23] Lixin Duan, Ivor W. Tsang, and Dong Xu. 2012. Domain Transfer Multiple Kernel Learning. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 34(3). 465–479.

[24] Kilian Fatras, Younes Zine, Szymon Majewski, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. 2021. Minibatch Optimal Transport Distances: Analysis and Applications. In [arXiv:2101.01792](https://arxiv.org/abs/2101.01792).

[25] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weber, Geoffrey I. Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Fleuret. 2019. InceptionTime: Finding AlexNet for Time-Series Classification. In [arXiv:1909.04939](https://arxiv.org/abs/1909.04939).

[26] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. 2013. Unsupervised Visual Domain Adaptation Using Subspace Alignment. In *Intl. Conf. on Computer Vision (ICCV)*. Sydney, Australia, 2960–2967.

[27] Sira Ferradans, Nicolas Papadakis, Gabriel Peyré, and Jean-François Aujol. 2013. Regularized Discrete Optimal Transport. In *Proc. Scale Space Variational Methods Comput. Vis.* 428–439.

[28] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T. H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. 2021. POT: Python Optimal Transport. In *Journal of Machine Learning Research (JMLR)*, Vol. 22. 1–8. <https://pythont.github.io/index.html>

[29] Rémi Flamary, Nicolas Courty, Alain Rakotomamonjy, and Devis Tuia. 2014. Optimal Transport with Laplacian Regularization. In *Advances in Neural Information Processing Systems (NIPS) Workshop on Optimal Transport and Machine Learning*. Montréal, Canada.

[30] Muhammad Ghifary, W. Bastiaan Kleijn, and Mengjie Zhang. 2014. Domain Adaptive Neural Networks for Object Recognition. In *Pacific Rim Intl. Conf. on Artificial Intelligence (PRICAI)*. Gold Coast, Australia.

[31] Boqing Gong, Kristen Grauman, and Fei Sha. 2014. Learning Kernels for Unsupervised Domain Adaptation with Applications to Visual Object Recognition. In *Intl. Journal of Computer Vision (IJCV)*, Vol. 109. 3–27.

[32] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. 2012. Geodesic Flow Kernel for Unsupervised Domain Adaptation. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Providence, RI, 2066–2073.

[33] Mehrtash Harandi, Mathieu Salzmann, and Fatih Porikli. 2014. Bregman Divergences for Infinite Dimensional Covariance Matrices. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Columbus, OH.

[34] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. 2019. Bag of Tricks for Image Classification with Convolutional Neural Networks. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, 558–567.

[35] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 97. 2790–2799.

[36] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-Tuning for Text Classification. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Vol. 1. Melbourne, Australia, 328–339.

[37] Hailin Hu, Mingjian Tang, and Chengcheng Bai. 2020. DATSING: Data Augmented Time-Series Forecasting with Adversarial Domain Adaptation. In *ACM Intl. Conf. on Information & Knowledge Management (CIKM)*. 2061–2064.

[38] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. 2006. Correcting Sample Selection Bias by Unlabeled Data. In *Advances in Neural Information Processing Systems (NIPS)*. 601–608.

[39] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2020. MHTN: Modal-Adversarial Hybrid Transfer Network for Cross-Modal Retrieval. In *Trans. on Cybernetics*, Vol. 50(3). 1047–1059. <https://doi.org/10.1109/TCYB.2018.2879846>

[40] Hal Daumé III. 2007. Frustratingly Easy Domain Adaptation. In *Association of Computational Linguistics (ACL)*. Prague, Czech Republic, 256–263.

[41] Junguang Jiang, Ximei Wang, Mingsheng Long, and Jianmin Wang. 2020. Resource Efficient Domain Adaptation. In *ACM Intl. Conf. on Multimedia (ACMMM)*. 2220–2228.

[42] Min Jiang, Wenzhen Huang, Zhongqiang Huang, and Gary G. Yen. 2015. Integration of Global and Local Metrics for Domain Adaptation Learning via Dimensionality Reduction. In *Trans. on Cybernetics*, Vol. 47(1). 38–51. <https://doi.org/10.1109/TCYB.2015.2502483>

[43] Yaoayong Jin, Youngsuk Park, Danielle C. Maddix, Hao Wang, and Yuyang Wang. 2021. Domain Adaptation for Time Series Forecasting via Attention Sharing. In [arXiv:2102.06828](https://arxiv.org/abs/2102.06828).

[44] L. V. Kantorovich. 2006. On the Translocation of Masses. In *Journal of Mathematical Sciences*, Vol. 133. 1381–1382.

[45] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. 2017. LSTM Fully Convolutional Networks for Time-Series Classification. In [arXiv:1709.05206](https://arxiv.org/abs/1709.05206).

[46] Andreas Klaß, Sven M. Lorenz, Martin W. Lauer-Schmalz, David Rügamer, Bernd Bischl, Christopher Mutschler, and Felix Ott. 2022. Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift. In [arXiv:2206.08640](https://arxiv.org/abs/2206.08640).

[47] Feng Liu, Guangquan Zhang, and Jie Lu. 2020. Heterogeneous Domain Adaptation: An Unsupervised Approach. In *Trans. on Neural Networks and Learning Systems*, Vol. 31(12). 5588–5602. <https://doi.org/10.1109/TNNLS.2020.2973293>

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

214

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

[48] Mingsheng Long, Yue Cao, Lianmin Wang, and Michael I. Jordan. 2015. Learning Transferable Features with Deep Adaptation Networks. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 37. 97–105.

[49] Mingsheng Long, Jianmin Wang, Guiguang Ding, Simo Jialin Pan, and Philip S. Yu. 2014. Adaptation Regularization: A General Framework for Transfer Learning. In *Trans. on Knowledge and Data Engineering*, Vol. 26(5).

[50] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S. Yu. 2013. Transfer Feature Learning with Joint Distribution Adaptation. In *Intl. Conf. on Computer Vision (ICCV)*. Sydney, Australia, 2200–2207. <https://doi.org/10.1109/ICCV.2013.274>

[51] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S. Yu. 2014. Transfer Joint Matching for Unsupervised Domain Adaptation. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Columbus, OH, 1410–1417. <https://doi.org/10.1109/CVPR.2014.183>

[52] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2016. Unsupervised Domain Adaptation with Residual Transfer Networks. In *Advances on Neural Information Processing Systems (NIPS)*. 136–144.

[53] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2017. Deep Transfer Learning with Joint Adaptation Networks. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 70. 2208–2217.

[54] Yong Luo, Tongliang Liu, Yonggang Wen, and Dacheng Tao. 2018. Online Heterogeneous Transfer Metric Learning. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, 2525–2531.

[55] Maher Moakher and Philipp G. Batchelor. 2006. Symmetric Positive-Definite Matrices: From Geometry to Applications and Visualization. In *Visualization and Processing of Tensor Fields. Mathematics and Visualization*, Springer, Berlin, Heidelberg.

[56] Trung Thanh Nguyen, Tomi Silander, Zhuoru Li, and Tze-Yun Leong. 2017. Scalable Transfer Learning in Heterogeneous, Dynamic Environments. In *Artificial Intelligence*, Vol. 247. 70–94. <https://doi.org/10.1016/j.artint.2015.09.013>

[57] Ignacio Oguiza. 2020. tsai - A State-of-the-art Deep Learning Library for Time-Series and Sequential Data. Github. <https://github.com/timeseriesAI/tsai>

[58] Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. Cross-Modal Common Representation Learning with Triplet Loss Functions. In [arXiv:2202.07901](https://arxiv.org/abs/2202.07901).

[59] Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In *Proc. of the IEEE/CVF Winter Conf. on Applications of Computer Vision (WACV)*. Waikoloa, HI, 266–276.

[60] Felix Ott, David Rügamer, Lucas Heublein, Tim Hamann, Jens Barth, Bernd Bischl, and Christopher Mutschler. 2022. Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In [arXiv:2202.07036](https://arxiv.org/abs/2202.07036).

[61] Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. 2020. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, Vol. 4(3), Article 92. Cancún, Mexico. <https://doi.org/10.1145/3411842>

[62] Simo Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. 2011. Domain Adaptation via Transfer Component Analysis. In *Trans. on Neural Networks*, Vol. 22(2). 199–210. <https://doi.org/10.1109/TNN.2010.2091281>

[63] Simo Jialin Pan and Qiang Yang. 2009. A Survey on Transfer Learning. In *Trans. on Knowledge and Data Engineering*, Vol. 22(10). 1345–1359. <https://doi.org/10.1109/TKDE.2009.191>

[64] Vishal M. Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2015. Visual Domain Adaptation: A Survey of Recent Advances. In *IEEE Signal Processing Magazine*, Vol. 32(3). 53–69. <https://doi.org/10.1109/MSP.2014.2347059>

[65] Karl Pearson. 1920. Notes on the History of Correlation. In *Biometrika*, Vol. 13(1). 25–45.

[66] Réjean Plamondon and Sargur N. Srihari. 2000. Online and Off-line Handwriting Recognition: A Comprehensive Survey. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 22(1). 63–84.

[67] Sanjay Purushotham, Wilka Carvalho, Tanachat Nilanon, and Yan Liu. 2017. Variational Recurrent Adversarial Deep Domain Adaptation. In *Intl. Conf. on Learning Representations (ICLR)*.

[68] Elahe Rahimian, Soheil Zabih, Seyed Farokh Atashzar, Amir Asif, and Arash Mommadi. 2019. XceptionTime: A Novel Deep Architecture based on Depthwise Separable Convolutions for Hand Gesture Classification. In [arXiv:1911.03803](https://arxiv.org/abs/1911.03803).

[69] Viresh Ranjan, Nikhil Rasiwasia, and C. V. Jawahar. 2015. Multi-Label Cross-Modal Retrieval. In *Intl. Conf. on Computer Vision (ICCV)*. Santiago de Chile, Chile, 4094–4102. <https://doi.org/10.1109/ICCV.2015.466>

[70] Esmat Rashedi, Hossein Nezamabadi-pour, and Saeid Saryazdi. 2013. A Simultaneous Feature Adaptation and Feature Selection Method for Content-based Image Retrieval Systems. In *Knowledge-Based Systems*, Vol. 39. 85–94.

[71] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. Adapting Visual Category Models to New Domains. In *Euro. Conf. on Computer Vision (ECCV)*. Vol. 6314. 213–226.

[72] Filippo Santambrogio. 2015. Optimal Transport for Applied Mathematicians - Calculus of Variations, PDEs, and Modeling. In *Progress in Nonlinear Differential Equations and Their Applications (PNLDE)*, Vol. 87.

[73] Sandeepkumar Satpal and Sunita Sarawagi. 2007. Domain Adaptation of Conditional Probability Models Via Feature Subsetting. In *Euro. Conf. on Principles of Data Mining and Knowledge Discovery (CPDMKD)*, Vol. 4702. 224–235.

[74] Ling Shao, Fan Zhu, and Xuelong Li. 2014. Transfer Learning for Visual Categorization: A Survey. In *Trans. on Neural Networks and Learning Systems*, Vol. 26(5). 1019–1034. <https://doi.org/10.1109/TNNLS.2014.2330900>

[75] Baochen Sun, Jiashin Feng, and Kate Saenko. 2016. Correlation Alignment for Unsupervised Domain Adaptation. In [arXiv:1612.01939](https://arxiv.org/abs/1612.01939).

[76] Baochen Sun and Kate Saenko. 2015. Subspace Distribution Alignment for Unsupervised Domain Adaptation. In *British Machine Vision Conf. (BMVC)*, Vol. 24. 10. <https://doi.org/10.5244/C.29.24>

[77] Baochen Sun and Kate Saenko. 2016. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Euro. Conf. on Computer Vision (ECCV)*, Vol. 9915. 443–450.

[78] Chang Wei Tan, Angus Dempster, Christoph Bergmeir, and Geoffrey I. Webb. 2021. MultiRocket: Multiple Pooling Operators and Transformations for Fast and Effective Time-Series Classification. In [arXiv:2102.00457](https://arxiv.org/abs/2102.00457).

[79] Wensi Tang, Guodong Long, Lu Liu, Tianyi Zhou, Jing Jiang, and Michael Blumenstein. 2020. Rethinking 1D-CNN for Time-Series Classification: A Stronger Baseline. In [arXiv:2002.10061](https://arxiv.org/abs/2002.10061).

[80] Benjamin Tardy, Jordi Inglada, and Julien Michel. 2019. Assessment of Optimal Transport for Operational Land-Cover Mapping Using High-Resolution Satellite Images Time Series Without Reference Data of the Mapping Period. In *MDPI Remote Sensing*, Vol. 11(9). <https://doi.org/10.3390/rs11091047>

[81] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep Domain Confusion: Maximizing for Domain Invariance. In [arXiv:1412.3474](https://arxiv.org/abs/1412.3474).

[82] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. In *Journal of Machine Learning Research (JMLR)*, Vol. 9(86). 2579–2605.

[83] Cédric Villani. 2008. Optimal Transport, Old and New. In *Springer*.

[84] Chang Wang and Sridhar Mahadevan. 2019. Manifold Alignment using Procrustes Analysis. In *MDPI Remote Sensing*, Vol. 11(9). <https://doi.org/10.3390/rs1109156>.

[85] Jindong Wang, Yiqiang Chen, Lisha Hu, Xiaohui Peng, and Philip S. Yu. 2017. Stratified Transfer Learning for Cross-Domain Activity Recognition. In [arXiv:1801.00820](https://arxiv.org/abs/1801.00820).

[86] Zhiqiang Wang, Weizhong Yan, and Tim Oates. 2016. Time-Series Classification from Scratch with Deep Neural Networks: A Strong Baseline. In [arXiv:1611.06455](https://arxiv.org/abs/1611.06455).

[87] Garrett Wilson, Janardhan Rao Doppa, and Diane J. Cook. 2020. Multi-Source Deep Domain Adaptation with Weak Supervision for Time-Series Sensor Data. In *ACM Intl. Conf. on Knowledge Discovery & Data Mining (SIGKDD)*. 1768–1778.

[88] Yuguang Yan, Qingyao Wu, Mingkui Tan, Michael K. Ng, Huaqing Min, and Ivor W. Tsang. 2017. Online Heterogeneous Transfer by Hedge Ensemble of Offline and Online Decisions. In *Trans. on Neural Networks and Learning Systems*, Vol. 29(7). 3252–3263.

[89] Liu Yang, Liping Jing, Jian Yu, and Michael K. Ng. 2015. Learning Transferred Weights From Co-Occurrence Data for Heterogeneous Transfer Learning. In *Trans. on Neural Networks and Learning Systems*, Vol. 27(11). 2187–2200. <https://doi.org/10.1109/TNNLS.2017.2751102>

[90] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A Transformer-based Framework for Multivariate Time-Series Representation Learning. In *ACM Intl. Conf. on Knowledge Discovery & Data Mining (SIGKDD)*. 2114–2124.

[91] Youshan Zhang. 2021. A Survey of Unsupervised Domain Adaptation for Visual Recognition. In [arXiv:2112.06745](https://arxiv.org/abs/2112.06745).

[92] Zhen Zhang, Mianzhi Wang, Yan Huang, and Arye Nehorai. 2018. Aligning Infinite-Dimensional Covariance Matrices in Reproducing Kernel Hilbert Spaces for Domain Adaptation. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT, 3437–3445. <https://doi.org/10.1109/CVPR.2018.00362>

[93] Peilin Zhao, Steven C. H. Hoi, Jialei Wang, and Bin Li. 2014. Online Transfer Learning. In *Research Collection School of Information Systems*, Vol. 216. 76–102.

[94] Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Simo Jialin Pan, and Qing He. 2018. Supervised Representation Learning with Double Encoding-Layer Autoencoder for Transfer Learning. In *ACM Trans. on Intelligent Systems and Technology (TIST)*, Vol. 9(2), Article 16. 1–17.

[95] Hank Hankui Zhuo and Qiang Yang. 2014. Action-Model Acquisition for Planning via Transfer Learning. In *Artificial Intelligence*, Vol. 212. 80–103. <https://doi.org/10.1016/j.artint.2014.03.004>

[96] Xiaowu Zou, Zidong Wang, Qi Li, and Weiguo Sheng. 2019. Integration of Residual Network and Convolutional Neural Network Along with Various Activation Functions and Global Pooling for Time-Series Classification. In *Neurocomputing*, Vol. 367. 39–45.

A APPENDICES

A.1 Notations

Table 5: Overview of notations used for our domain adaptation methodology.

Notation	Description
Multivariate Time-Series (MTS) Classification	
$U = \{u_1, \dots, u_m\} \in \mathbb{R}^{m \times l}$	An MTS (ordered sequence) of $l \in \mathbb{N}$ streams
$u_i = (u_{i,1}, \dots, u_{i,l})$	A stream of an MTS
$i \in \{1, \dots, m\}$	
$m \in \mathbb{N}$	Length of a time-series
$\mathcal{U} = \{U_1, \dots, U_{n_U}\}$	Set of n_U time-series with $\mathcal{U} \in \mathbb{R}^{n_U \times m \times l}$
$y \in \mathcal{Y}$	Unknown class label for a given MTS
\mathcal{Y}	Label space
$ \mathcal{Y} $	Number of class labels
Domain Adaptation	
U_T	Target domain dataset
U_S	Source domain dataset
\mathcal{U}_{T_t}	Training subset of the target domain dataset
\mathcal{U}_{T_v}	Validation subset of the target domain dataset
\mathcal{U}_{S_t}	Training subset of the source domain dataset
\mathcal{U}_{S_v}	Validation subset of the source domain dataset
U_T	MTS of the target domain dataset
U_S	MTS of the source domain dataset
U_{T_t}	Trainings subset of the target domain dataset
U_{T_v}	Validation subset of the target domain dataset
U_{S_t}	Trainings subset of the source domain dataset
U_{S_v}	Validation subset of the source domain dataset
μ_T	Mean of \mathcal{U}_T
μ_S	Mean of \mathcal{U}_S
C_T	Covariance matrix of \mathcal{U}_T
C_S	Covariance matrix of \mathcal{U}_S
$T: \mathcal{D}_S \rightarrow \mathcal{D}_T$	Transformation
$f(U_{T_t}) \in \mathbb{R}^{q_T \times w_T}$	Latent embedding of the training MTS of the target domain dataset
$f(U_{S_t}) \in \mathbb{R}^{q_S \times w_S}$	Latent embedding of the training MTS of the source domain dataset
q_*, w_*	Size of the embedding $f(U_{*}) \in \mathbb{R}^{q_* \times w_*}$
X	Feature space
$P(X)$	Marginal probability of X
$P(X, Y)$	Joint distribution of X and Y
$P(Y X)$	Conditional distribution between X and Y
\mathcal{D}	Domain
$\mathcal{D}_T = \{X_T^i, Y_T^i\}_{i=1}^{N_T}$	Target domain of N_T labeled samples
$\mathcal{D}_S = \{X_S^i, Y_S^i\}_{i=1}^{N_S}$	Source domain of N_S labeled samples
\mathcal{H}	Hypothesis space
\mathcal{R}_T	Target domain error
\mathcal{R}_S	Source domain error
$d_{\mathcal{H} \setminus \mathcal{H}}$	Discrepancy distance between \mathcal{D}_S and \mathcal{D}_T w.r.t. the hypothesis space \mathcal{H}
Optimal Transport	
$C(T)$	Transportation cost
$c: \mathcal{D}_T \times \mathcal{D}_S \rightarrow \mathbb{R}^+$	Distance function
T_0	Optimal transportation problem
$\alpha \in \Theta$	General coupling
$\Theta \in P(\mathcal{D}_T, \mathcal{D}_S)$	Set of all probabilistic couplings
W_p	Wasserstein distance of order p
$c(U_T, U_S)$	Cost function
$d(U_T, U_S)^p$	Distance function of order p

A.2 Correlation Alignment (CORAL)

The calculation of subspace-based methods is simple and efficient. Hence, we use CORAL [75], which minimizes the domain shift by aligning the second-order statistics (i.e., the original feature distributions of source and target domains). Suppose μ_T and μ_S are the means of $f(U_T)$ and $f(U_S)$, and C_T and C_S are the covariance matrices. We normalize the features to have zero mean ($\mu_T = \mu_S = 0$). Then, CORAL minimizes the distance between $f(U_T)$ and $f(U_S)$ by a linear transformation \mathbf{A} by

$$\min_{\mathbf{A}} \|C_S - C_T\|_F^2 = \min_{\mathbf{A}} \|\mathbf{A}^T C_T \mathbf{A} - C_T\|_F^2, \quad (9)$$

where $\|\cdot\|_F^2$ is the squared Frobenius norm, and C_S is the covariance of the transformed source features $f(U_S)\mathbf{A}$ [75]. The optimal solution of this problem is given by

$$\mathbf{A}^* = \mathbf{P}_S \Sigma_S^{+ \frac{1}{2}} \mathbf{P}_S^T \mathbf{P}_{T[1:r]} \Sigma_{T[1:r]}^{\frac{1}{2}} \mathbf{P}_{T[1:r]}^T, \quad (10)$$

with $r = \min(r_{C_S}, r_{C_T})$, where r_{C_S} and r_{C_T} denote the rank of C_S and C_T , and Σ^+ is the Moore-Penrose pseudoinverse of Σ . We use the singular value decomposition of a real matrix Y to compute the largest $r \leq r_Y$ singular values $\Sigma_{Y[1:r]}$, and left and right singular vectors $\mathbf{P}_{Y[1:r]}$ and $\mathbf{V}_{Y[1:r]}$ of $Y = \mathbf{P}_Y \Sigma_Y \mathbf{V}_Y$ of the real matrix Y of rank r_Y [7].

A.3 Datasets Overview

Table 6 gives an overview of sample counts for the right-handed target domains for training \mathcal{U}_{T_t} and validation \mathcal{U}_{T_v} . Results for these datasets are given in Table 2 and 3. Here, the left-handed dataset comprise all writers. Table 7 shows the sample numbers for left-handed writers separated for each writer at an 80/20 training validation split. We propose counts for the source domains for training \mathcal{U}_{S_t} and validation \mathcal{U}_{S_v} data. Results are averaged over all writers (four for the OnHW-symbols and split OnHW-equations datasets, and nine for the OnHW-chars dataset) and are given in Table 4 for domain adaptation and in Table 1 for transfer learning.

Table 6: Overview of sample numbers of online handwriting (OnHW) recognition datasets for writer-dependent (WD) and writer-independent (WI) and right- and left-handed classification tasks. Top line: training. Bottom line: validation.

	Dataset	Right-handed		Left-handed	
		WD	WI	WD	WI
OnHW-symbols [60]		1,853	1,715	288	267
		473	611	73	94
Split OnHW-equations [60]		31,697	30,408	5,021	4,579
		7,946	9,235	1,259	1,701
OnHW-chars [61]	lower	11,524	11,647	903	781
		4,101	3,978	218	368
	upper	11,542	11,672	925	757
		4,108	3,978	224	364
	combined	23,066	23,319	1,821	1,538
		8,209	7,956	449	732

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

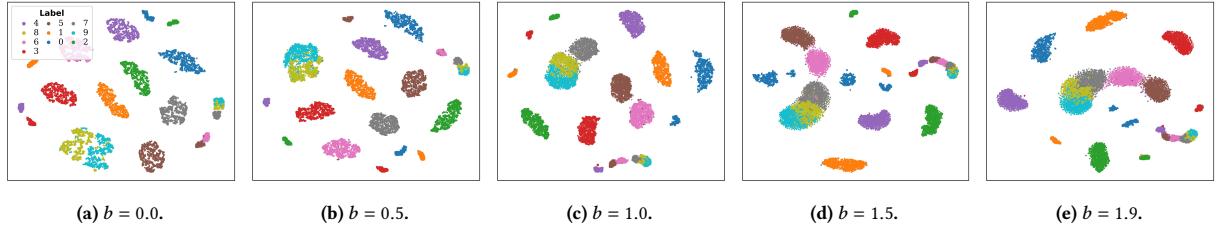


Figure 9: Embedding visualization for the sinusoidal datasets for the target domain \mathcal{U}_T and the source domain \mathcal{U}_{S_v} with noise parameters $b \in B = \{0.0, 0.5, 1.0, 1.5, 1.9\}$. Marker \cdot : target domain embeddings. Marker \times : source domain validation embeddings.

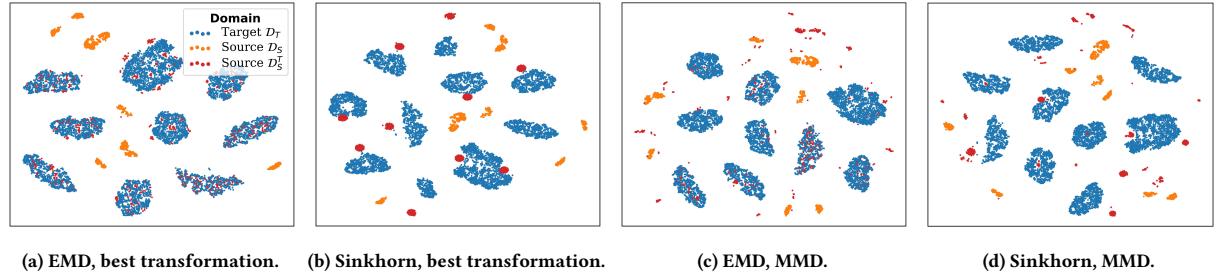


Figure 10: Embedding visualization for the target domain \mathcal{U}_T , source domain \mathcal{U}_S and transformed source domain \mathcal{U}_S^T for the sinusoidal datasets. We show embeddings for EMD and Sinkhorn transport without regularization. Evaluated are nine metrics for transformation selection: CC, PC, CORAL (standard, Jeff and Stein), correlation normalized, MMD, kMMD, HoMM, without transformation and best transformation. Noise is $b = 0.5$. Note that we applied t-SNE for each plot separately that leads to different embeddings for the same data (i.e., target domain \mathcal{D}_T). Figure 11 follows.

Table 7: Overview of number of samples for each left-handed writer for the OnHW datasets for train/validation splits. For OnHW-chars, we count for lower/upper/combined.

Dataset	Writer	Train (\mathcal{U}_{S_I})	Val. (\mathcal{U}_{S_v})
OnHW-symbols-L [60]	1	71	18
	2	72	18
	3	70	18
	4	75	19
Split OnHW-equations-L [60]	1	1,299	327
	2	1,067	267
	3	1,295	324
	4	1,360	341
OnHW-chars-L [61]	1	78 / 89 / 167	26 / 15 / 41
	2	78 / 89 / 167	26 / 15 / 41
	3	78 / 70 / 148	26 / 11 / 37
	4	78 / 89 / 167	26 / 15 / 41
	5	79 / 89 / 168	26 / 15 / 41
	6	202 / 218 / 420	62 / 42 / 104
	7	78 / 89 / 167	26 / 15 / 41
	8	78 / 89 / 167	26 / 15 / 41
	9	120 / 130 / 250	360 / 26 / 386

A.4 Feature Embeddings for the Sinusoidal Dataset

To better visualize the differences between domain-dependent embeddings and different methods, we plot two dimensional embeddings of the features $f(\mathbf{U})$. For the sinusoidal dataset, $f(\mathbf{U})$ is of size 50×30 (reshaped 1,500). We use t-SNE [82] with initial dimension 1,500, perplexity of 30, an initial momentum of 0.5, and a final momentum of 0.8. Figure 9 visualizes the feature embeddings for different noise parameters b with class label dependent colors. Without noise ($b = 0$), the clusters are clearly separable, while only label '8' and '9' overlap. As the noise increases $b = 1.5$, also the cluster with label '7' overlaps, and finally label '6' and '3' for $b = 1.9$. This is reflected by the results in Figure 8, where the accuracy drops for $b > 0.7$. The validation samples with domain shift are notably distant to the target samples.

Figure 10 and 11 visualize the feature embeddings $f(\mathbf{U})$ for \mathcal{U}_T (blue), \mathcal{U}_S (orange), and the transformed \mathcal{U}_S^T (red). It is notable that EMD forms spread clusters of the specific sample embedding for the transformed source domain, while Sinkhorn shapes small clusters that are close to the target domain clusters, but are outlying. This distance increases for a higher noise. Choosing the best transformation (Figure 10a and 10b), the transformed embeddings are mostly consistent, the kMMD (Figure 11a and 11b) and cross correlation (Figure 11e and 11f) distance metrics (that result in the

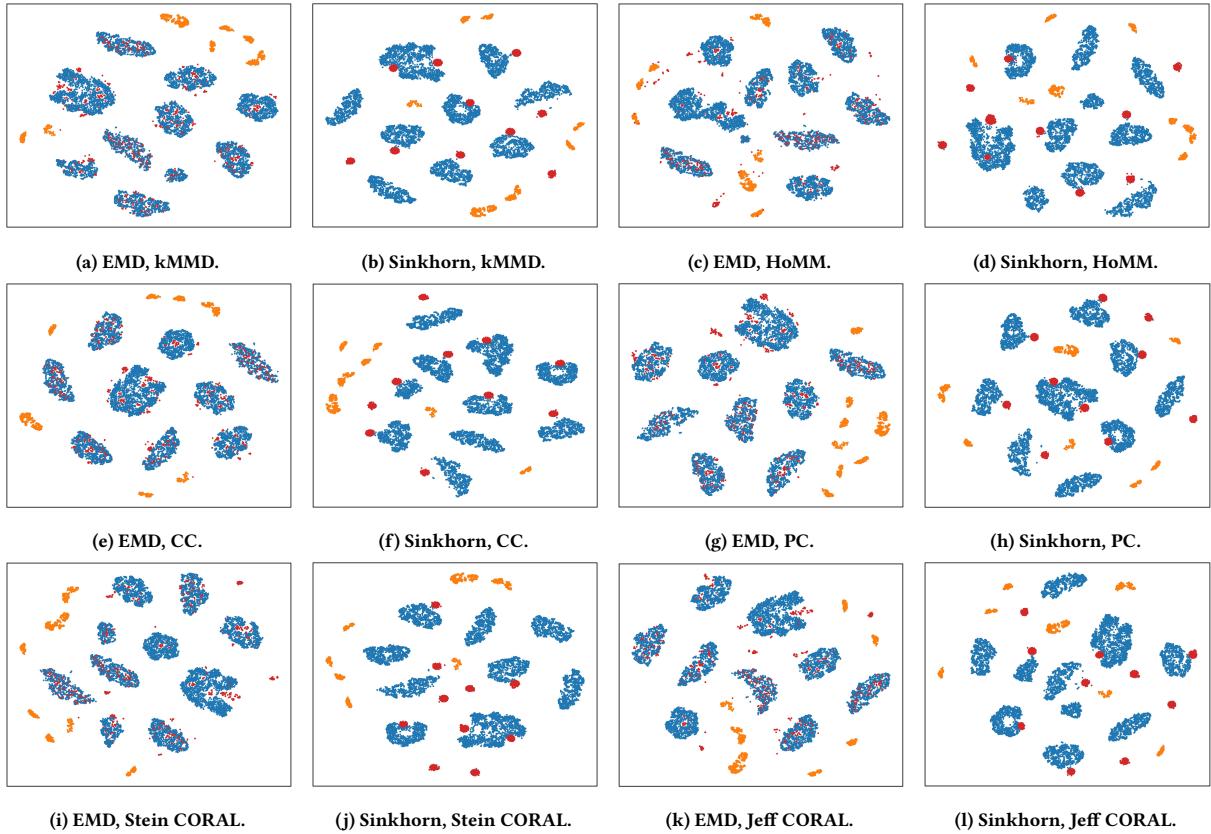


Figure 11: For the caption, see Figure 10.

highest classification accuracy), are similar to the best transformation. HoMM (Figure 11c and 11d) with the lowest classification accuracy, also results in a larger spread of transformed clusters. Different are the variances Stein and Jeff CORAL trained with EMD where the transformed source domain samples are notably out-of the target domain samples (see Figure 11i and 11k).

A.5 Deep Metric Learning Searches

We perform a large hyperparameter search on the OnHW-symbols dataset for all optimal transport techniques. Results are shown in Figure 12. The differences between EMD and SEMD are marginally, as well as between L_pL_1 and L_1L_2 regularization techniques. In general Sinkhorn performs better than EMD. While the *log* metric performs best for EMD, *median*, *max* and *loglog* yield better results for Sinkhorn. The differences for distance metrics are marginal for Sinkhorn. For follow-up trainings, we choose the squared Euclidean distance metric with *loglog* metric. Indeed, the similarity comparison method for transformation selection has the highest impact. kMMD consistently yields the highest classification accuracies followed by Stein and Jeff CORAL. Again, HoMM of order 3 can outperform MMD, but not kMMD.

A.6 Evaluation per Writer

We adapt each writer separately as writer can have very different writing styles, and hence, different domains of sensor features. Figure 13 to Figure 17 shows all results for five different optimal transport techniques and transformation selection methods. Using the best transformation mostly achieves 100%, which is the upper bound for the optimal transformation. Without transformation, the right-handed model leads to a poor accuracy of below 10% (the lower bound). For the OnHW-symbols and split OnHW-equations datasets, the accuracy of the models drop for the writer with ID 3. For the OnHW-chars dataset, the writers with ID 3 and 6 are outlier. The reason is that these persons wrote inconsistent. Again, the difference between EMD and SEMD, and Sinkhorn with and without regularization is marginal in accuracy.

Figure 18 shows results of transfer learning separated for all writers for all OnHW datasets. Consistently, adapting an additional layers yields the lowest classification results. While the OnHW-symbols dataset is rather small, all post-training techniques fail to successfully classify symbols (see Figure 18a). The split OnHW-equations dataset is large, and hence, post-training results in classification accuracies between 60% and 100% (see Figure 18b). Results

7. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift 218

MM '22, October 10–14, 2022, Lisboa, Portugal

Felix Ott et al.

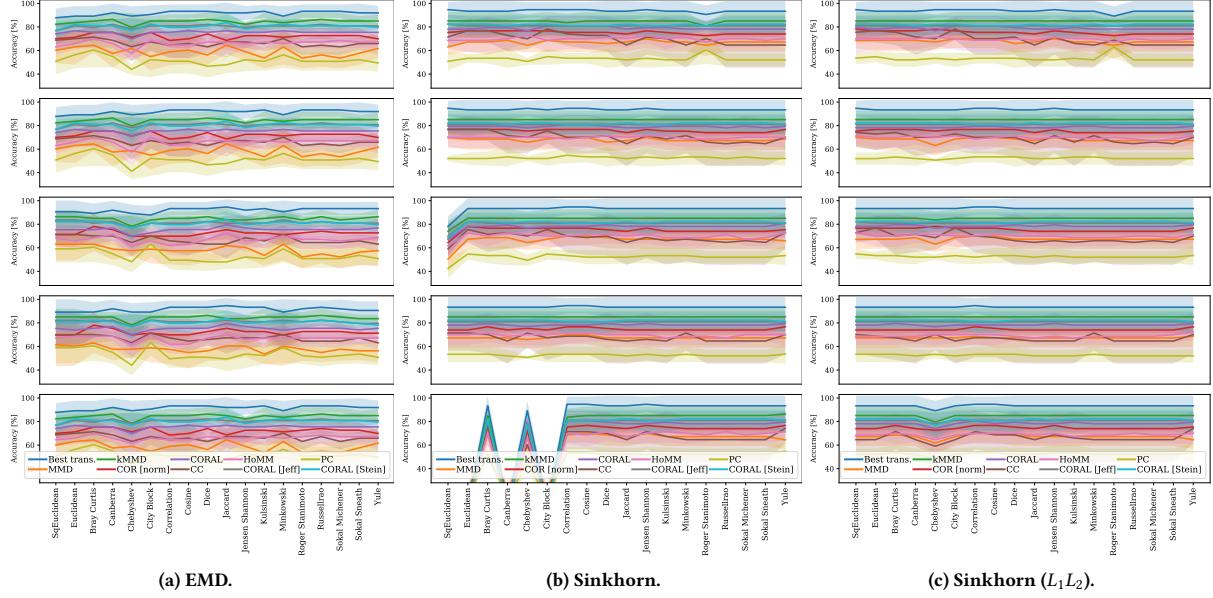


Figure 12: Hyperparameter search for distance metrics for optimal transport methods and transformation selection methods on the OnHW-symbols dataset. Results are averaged over four writers. 1: median. 2: max. 3: log. 4: loglog. 5: None.

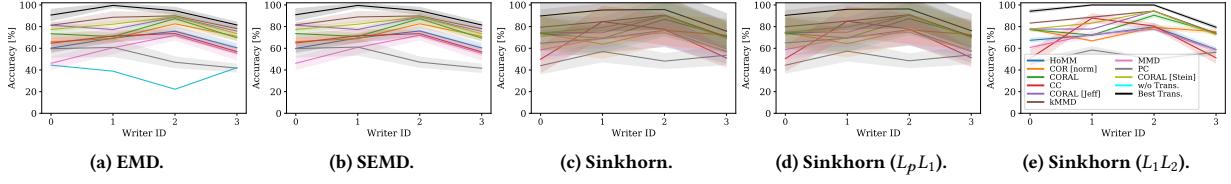


Figure 13: Evaluation of the transformed embeddings for the left-handed OnHW-symbols dataset for each of the four writers.

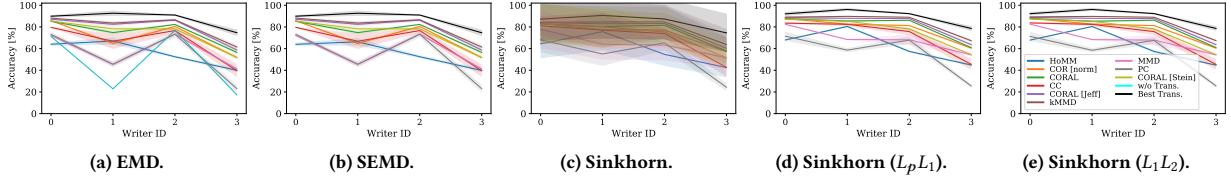


Figure 14: Evaluation of the transformed embeddings for the left-handed split OnHW-equations dataset for each of the four writers.

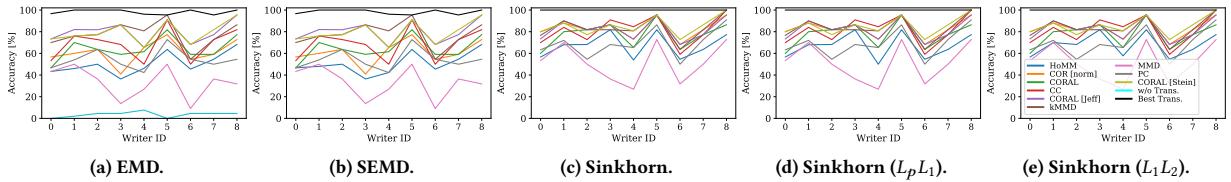


Figure 15: Evaluation of the transformed embeddings for the left-handed OnHW-chars (lower) dataset for each writer.

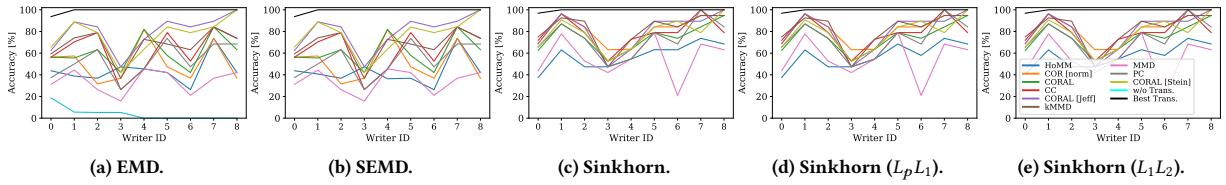


Figure 16: Evaluation of the transformed embeddings for the left-handed OnHW-chars (upper) dataset for each writers.

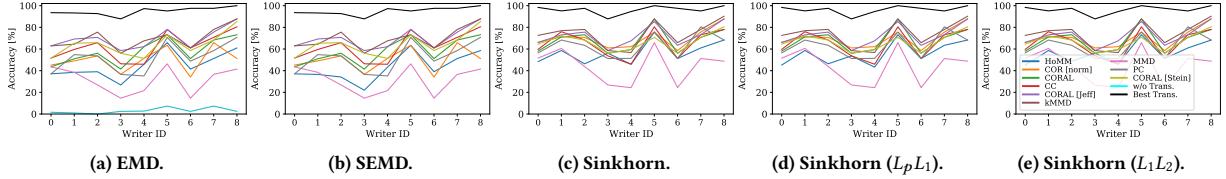


Figure 17: Evaluation of the transformed embeddings for the left-handed OnHW-chars (combined) dataset for each writer.

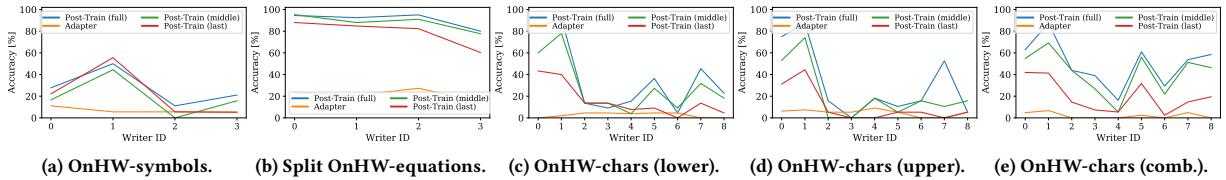


Figure 18: Evaluation of transfer learning for the OnHW datasets for each of the four, respectively nine, writers.

Table 8: Architecture details for the sinusoidal datasets.

Layer	Features
Encoder	
Input	Size: (Sequence length = 200, 13)
Convolution (1D)	Filters: 50, kernel size: 4, activation: <i>softmax</i>
Max Pooling (1D)	Pool size: 4
Classifier	
Batch Normalization	
Dropout	Rate 20%
LSTM	Units: 10, activation: <i>tanh</i>
Dense	Units: 20
Output	Size: <i>number classes</i> (10)

for the OnHW-chars lower, upper and combined are similar (see Figure 18c to 18e). Post-training the full model yields marginally better results than training only the last layer or the layers after the middle of the model. Hence, the model requires to overfit on the specific writer. Results highly vary with the writer ID.

A.7 Details on the Architectures

Table 8 and 9 show architecture details for the sinusoidal dataset, and OnHW datasets, respectively. Both models contain a feature extractor of the time-series datasets, and temporal unit, and *dense* layers for classification. We use a small LSTM of 10 units for the synthetic dataset, and two stacked bidirectional LSTM layers of 60 units each for the OnHW datasets. For the optimal transport methods, we use the output of the *max pooling* layer before the *batch normalization* and *dropout* layers and the temporal units. We

Table 9: Architecture details for the OnHW datasets.

Layer	Features
Encoder	
Input	Size: (Sequence length, 1)
Convolution (1D)	Filters: 200, kernel size: 4, activation: <i>relu</i>
Max Pooling (1D)	Pool size: 2
Batch Normalization	
Dropout	Rate: 20%
Convolution (1D)	Filters: 200, kernel size: 4, activation: <i>relu</i>
Max Pooling (1D)	Pool size: 2
Classifier	
Batch Normalization	
Dropout	Rate: 20%
BiLSTM	Units: 60, activation: <i>tanh</i> , return sequ.: True
BiLSTM	Units: 60, activation: <i>tanh</i> , return sequ.: True
Dense	Units: 100
Dense (time distributed)	Units: <i>number classes</i> , activation: <i>softmax</i>
Output	Size: <i>number classes</i>

train the synthetic dataset for 100 epochs, a batch size of 100, the Adam optimizer with learning rate 0.0001, and the categorical cross-entropy loss. We train the OnHW datasets for 1,000 epochs, a batch size of 50, the Adam optimizer with learning rate 0.0001, and the categorical cross-entropy loss. For transfer learning techniques we train the synthetic dataset for 80 epochs, and the OnHW datasets for 100 epochs with the same optimizer parameters.

Chapter 8

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition

Contributing Article

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition. In *IEEE Access*, volume 11, pages 94148-94172, August 2023. doi:10.1109/ACCESS.2023.3310819.

Publisher Website

Paper available at: <https://ieeexplore.ieee.org/document/10235987>
Attached version: *arXiv preprint arXiv:2202.07901v2 [cs.LG]*

Copyright License

This work is licensed to IEEE under the Creative Commons Attribution-NonCommercial-No Derivatives 4.0 License (CCBY-NC-ND) (<https://creativecommons.org/licenses/by-nc-nd/4.0/>).

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing,

visualization, and project administration. David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Lucas Heublein contributed for the methodology (i.e., creation of models), software, validation, investigation, and data curation (i.e., data management). Bernd Bischl contributed by supervision. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), visualization (i.e., presentation of the published work), supervision, and funding acquisition of the project “Schreibtrainer” (grant number 16SV8228) by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

Datasets and Source Code

This paper uses the OnHW-words500 and OnHW-wordsRandom datasets from right-handed writers and left-handed writers, as well as the publicly available IAM-OffDB dataset. The synthetically generated sinusoidal curves and the corresponding Gramian angular summation field can be reproduced with the published source code. The source code to reproduce evaluation results is publicly available at:

<https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>

Statement about Recent, Related Research¹⁷

For the contributing paper (Ott et al., 2023c), we employed ScrabbleGAN (Fogel et al., 2020) as a generative adversarial network (GAN) for augmenting the offline handwriting data with a corpus of a diverse range of word labels. ScrabbleGAN was selected based on its suitability for real-world applications, as it is capable of generating realistic handwriting styles with high data fidelity and low inference times. Furthermore, it is capable of generating images of words with varying lengths, making it a versatile option for data augmentation purposes. The utilized data augmentation with generated offline handwriting data can be improved by recent generative models, which include the approaches presented in Mattick et al. (2021); Kang et al. (2021); Luo et al. (2022); Gan et al. (2023). The generator proposed by Kang et al. (2021) conditions on both visual appearance and textual content, and it can produce text-line samples with diverse handwriting styles that visually outperform ScrabbleGAN. On the other hand, HiGAN+ (Gan et al., 2023) introduces a contextual loss to enhance style consistency and achieves better calligraphic style transfer. HiGAN+ also reuses the writer identifier for style encoding, resulting in better evaluation metrics than ScrabbleGAN. Both models are recent generative models that offer an alternative to improve data augmentation with generated offline handwriting data.

Similarly, recent research provides alternatives that may improve classification results for offline handwriting recognition. The Gated-CNN-BGRU model proposed by de Sousa Neto et al. (2022) aims to achieve high performance with few training data while keeping the number of trainable parameters low. PARSeq_A (Bautista & Atienza, 2022) is an

ensemble of internal autoregressive language models with shared weights, which is learned through permutation language modeling. Meanwhile, recent developments in Transformer architectures have led to several Transformer-based models that have shown improved performance on optical character recognition datasets. Examples of such models include TrOCR (Li et al., 2023), MaskOCR (Lyu et al., 2022), and the method proposed by Diaz et al. (2021). While the contributing paper applied interline spacing reduction via seam carving, resizing the images to 50% height, and random projective (rotating and resizing lines) and random elastic transform, Chaudhary & Bali (2022) introduce an alternative data augmentation technique called *tiling and corruption*. This method involves dividing the input image into multiple small tiles of equal size, which are then replaced with corrupted tiles. Although the tiling and corruption augmentation technique reduces the state-of-the-art results (CER) on the IAM-OffDB dataset, it achieves comparable results with Jangpangi et al. (2022). However, pre-training with synthetic data (Li et al., 2023) and using Transformer-based architectures (Diaz et al., 2021) achieve lower CERs.

A survey of contrastive and triplet learning techniques and CMR methods is provided in Deldari et al. (2022a), which also presents a comparison between the method proposed in the contributing paper (Ott et al., 2023c) and other CMR methods. CMT-Co (Zhang et al., 2023) is a word-level contrastive learning technique that generated artifacts by moving characters within a word. The model emphasized the text content by using the moving direction and distance as supervision.

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition

FELIX OTT, Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Germany, LMU Munich, Germany, and Munich Center for Machine Learning (MCML), Germany

DAVID RÜGAMER, LMU Munich, Germany, Technical University of Dortmund, Germany, and Munich Center for Machine Learning (MCML), Germany

LUCAS HEUBLEIN, Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Germany

BERND BISCHL, LMU Munich, Germany and Munich Center for Machine Learning (MCML), Germany

CHRISTOPHER MUTSCHLER, Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Germany

Cross-modal representation learning learns a shared embedding between two or more modalities to improve performance in a given task compared to using only one of the modalities. Cross-modal representation learning from different data types – such as images and time-series data (e.g., audio or text data) – requires a deep metric learning loss that minimizes the distance between the modality embeddings. In this paper, we propose to use the triplet loss, which uses positive and negative identities to create sample pairs with different labels, for cross-modal representation learning between image and time-series modalities (CMR-IS). By adapting the triplet loss for cross-modal representation learning, higher accuracy in the main (time-series classification) task can be achieved by exploiting additional information of the auxiliary (image classification) task. Our experiments on synthetic data and handwriting recognition data from sensor-enhanced pens show improved classification accuracy, faster convergence, and better generalizability.

CCS Concepts: • **Hardware** → Emerging technologies; • Computing methodologies → Learning latent representations; Feature selection; Transfer learning; • **Human-centered computing** → Interactive systems and tools.

Additional Key Words and Phrases: online handwriting recognition, sequence-based learning, optical character recognition, sensor-enhanced pen, cross-modal retrieval, contrastive learning, triplet learning

ACM Reference Format:

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2023. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition. *J. ACM* 00, 0, Article 000 (January 2023), 40 pages. <https://doi.org/XXXXXX.XXXXXXX>

Authors' addresses: Felix Ott, felix.ott@iis.fraunhofer.de, Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Nordostpark 84, Nuremberg, 90411, Germany and LMU Munich, Ludwigstraße 33, Munich, 80539, Germany and Munich Center for Machine Learning (MCML), Ludwigstraße 33, Munich, 80539, Germany; David Rügamer, david.ruegamer@stat.uni-muenchen.de, LMU Munich, Ludwigstraße 33, Munich, 80539, Germany and Technical University of Dortmund, August-Schmidt-Straße 1, Dortmund, 44227, Germany and Munich Center for Machine Learning (MCML), Ludwigstraße 33, Munich, 80539, Germany; Lucas Heublein, heublein@iis.fraunhofer.de, Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Germany; Bernd Bischl, bernd.bischl@stat.uni-muenchen.de, LMU Munich, Ludwigstraße 33, Munich, 80539, Germany and Munich Center for Machine Learning (MCML), Ludwigstraße 33, Munich, 80539, Germany; Christopher Mutschler, christopher.mutschler@iis.fraunhofer.de, Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits, Nordostpark 84, Nuremberg, 90411, Germany.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2023 Association for Computing Machinery.

0004-5411/2023/1-ART000 \$15.00

<https://doi.org/XXXXXX.XXXXXXX>

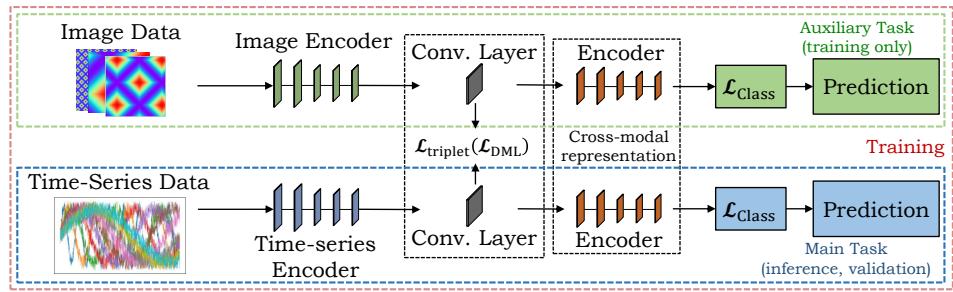


Fig. 1. **Method overview:** Cross-modal representation learning between image and time-series data using the triplet loss based on metric learning functions to improve the time-series classification task.

1 INTRODUCTION

Cross-modal retrieval (CMR) such as cross-modal representation learning [79] for learning across two or more modalities (i.e., image, audio, text and 3D data) has recently garnered substantial interest from the machine learning community. CMR can be applied in a wide range of applications, such as multimedia management [59] and identification [88]. Extracting information from several modalities and adapting the domain with cross-modal learning allows using the information in all domains [85]. Cross-modal representation learning, however, remains challenging due to the *heterogeneity gap* (i.e., inconsistent representation forms of different modalities) [49].

A limitation of cross-modal representation learning is that many approaches require the availability of all modalities at inference time. Image-to-caption CMR methods solve this via a separate encoder [19, 32]. However, in many applications, certain data sources are only available during training by means of elaborate laboratory setups [63]. For instance, consider a human pose estimation task that uses inertial sensors together with color videos during training, where a camera setup might not be available at inference time due to bad lighting conditions or other application-specific restrictions. Here, a model that allows inference on only the main modality is required, while auxiliary modalities may only be used to improve the training process (as they are not available at inference time) [42]. *Learning using privileged information* [102] is one approach in the literature that describes and tackles this problem. During training, in addition to X , it is assumed that additional *privileged information* X^* is available. However, this *privileged information* is not present in the inference stage [70].

For cross-modal representation learning, we need a deep metric learning technique that aims to transform training samples into feature embeddings that are close for samples that belong to the same class and far apart for samples from different classes [112]. As deep metric learning requires no model update (simply fine-tuning for training samples of new classes), deep metric learning is an often applied approach for continual learning [29]. Typical deep metric learning methods use not only simple distances (e.g., Euclidean distance), but also highly complex distances (e.g., canonical correlation analysis [85] and maximum mean discrepancy [68]). While cross-modal representation learning learns representations from all modalities, single-modal learning commonly uses pair-wise learning. The triplet loss [90] selects a positive and negative triplet pair for a corresponding anchor and forces the positive pair distance to be smaller than the negative pair distance. While research of triplet selection for single-modal classification is very advanced [25, 29, 42, 52, 55, 61, 73, 103, 106], pair-wise selection for cross-modal representation learning has mainly been investigated for specific applications [59, 121, 124], i.e., visual semantic embeddings [8, 19, 28, 84].

One exemplary application for cross-modal learning is handwriting recognition (HWR), which can be categorized into offline and online HWR. Offline HWR – such as optical character recognition (OCR) – concerns only analysis of the visual representation of handwriting and cannot be applied

for real-time recognition applications [33]. In contrast, online HWR works on different types of spatio-temporal signals and can make use of temporal information, such as writing speed and direction [81]. As an established real-world application of online HWR, many recording systems make use of a stylus pen together with a touch screen surface [3]. There also exist prototypical systems for online HWR when writing on paper [20, 27, 89, 108], but these are not yet suitable for real-world applications. However, a novel sensor-enhanced pen based on inertial measurement units (IMUs) may enable new online HWR applications for writing on normal paper. This pen has previously been used for single character [56, 73, 75, 77] and sequence [76] classification. However, the accuracy of previous online HWR methods is limited, due to the following reasons: (1) The size of datasets is limited, as recording larger amounts of data is time-consuming. (2) Extracting important spatio-temporal features is important. (3) Training a writer-independent classifier is challenging, as different writers can have notably different writing styles. (4) Evaluation performance drops for under-represented groups, i.e., left-handed writers. (5) The model overfits to seen words that can be addressed with generated models. A possible solution is to combine datasets of different modalities using cross-modal representation learning to increase generalizability. In this work, we combine offline HWR from generated images (i.e., OCR) and online HWR from sensor-enhanced pens by learning a common representation between both modalities. The aim is to integrate information on OCR – i.e., typeface, cursive or printed writing, and font thickness – into the online HWR task – i.e., writing speed and direction [104].

Our Contribution. Models that use rich data (e.g., images) usually outperform those that use a less rich modality (e.g., time-series). We therefore propose to train a shared representation using the triplet loss between pairs of image and time-series data to learn a cross-modal representation between both modality embeddings (cf. Figure 1). This allows for improving the accuracy of single-modal inference in the main task. Cross-modal learning between images and time-series data is rare. Furthermore, we propose a novel dynamic margin for the triplet loss based on the Edit distance. We prove the efficacy of our metric learning-based triplet loss for cross-modal representation learning both with simulated data and in a real-world application. More specifically, our proposed cross-modal representation learning technique 1) improves the multivariate time-series classification accuracy and convergence, 2) results in a small time-series-only network independent from the image modality while allowing for fast inference, and 3) has better generalizability and adaptability [49]. Our approach shows that the recent methods ScrabbleGAN [35] and OrigamiNet [117] are applicable in the real-world setup of offline HWR to enhance the online HWR task. We provide an extensive overview and technical comparison of related methods. Code and datasets are available upon publication.¹

The paper is organized as follows. Section 2 discusses related work followed by the mathematical foundation of our method in Section 3. The methodology is described in Section 4 and the results are discussed in Section 5.

2 RELATED WORK

In this section, we discuss related work – particularly, methods of offline HWR (in Section 2.1) and online HWR (in Section 2.2). We summarize approaches for learning a cross-modal representation from different modalities (in Section 2.3), pairwise and triplet learning (in Section 2.4), and deep metric learning (in Section 2.5) to minimize the distance between feature embeddings.

2.1 Offline Handwriting Recognition

In the following, we give a brief overview of offline HWR methods to select a suitable lexicon and language model-free method. For an overview of offline and online HWR datasets, see [50, 81]. For a

¹Code and datasets: <https://www.iis.fraunhofer.de/de/ff/lv/dataanalytics/anwproj/schreibtrainer/onhw-dataset.html>

more detailed overview, see Table 7 in the Appendix A.2. Methods for offline HWR range from hidden Markov models (HMMs) – such as [7, 30, 31, 60, 78] – to deep learning techniques that became predominant in 2014, such as convolutional neural networks (CNNs) as the methods by [82, 109]. The gated text recognizer [118] aims to automate the feature extraction from raw input signals with a minimum required domain knowledge. The fully convolutional network without recurrent connections is trained with the CTC loss. Thus, the gated text recognizer module can handle arbitrary input sizes and can recognize strings with arbitrary lengths. This module has been used for OrigamiNet [117] which is a segmentation-free multi-line or full-page recognition system. OrigamiNet yields state-of-the-art results on the IAM-OffDB dataset, and shows improved performance of gated text recognizer over VGG and ResNet26. Hence, we use the gated text recognizer module as our visual feature encoder for offline HWR. Furthermore, temporal convolutional networks (TCNs) employ the temporal context of the handwriting – such as the methods [92, 93]. More prominent became recurrent neural networks (RNNs) including long short-term memories (LSTMs), bidirectional LSTMs (BiLSTMs) [23, 51, 69, 98], and multidimensional RNNs [9, 10, 17, 21, 39, 58, 105]. Sequential architectures are perfect to fit text lines, due to the probability distributions over sequences of characters, and due to the inherent temporal aspect of text [54]. [38] introduced the BiLSTM layer in combination with the CTC loss. [80] showed that the performance of LSTMs can be greatly improved using dropout. GCRNN [11] combines a convolutional encoder (aiming for generic and multilingual features) and a BiLSTM decoder predicting character sequences. Additionally, [83] proposed a CNN+BiLSTM architecture that uses the CTC loss. Further methods that combine CNNs with RNNs are [62, 97, 115], while BiLSTMs are utilized in [16, 100].

Recent methods are generative adversarial networks (GANs) and Transformers. The first approach by [37] was a method to synthesize online data based on RNNs. The technique HWGAN by [53] extends this method by adding a discriminator \mathcal{D} . DeepWriting [1] is a GAN that is capable of disentangling style from content and thus making digital ink editable. [43] proposed a method to generate handwriting based on a specific author with learned parameters for spacing, pressure, and line thickness. [4] used a BiLSTM to obtain an embedding of the word to be rendered and added an auxiliary network as a recognizer \mathcal{R} . The model is trained with a combination of an adversarial loss and the CTC loss. ScrabbleGAN by [35] is a semi-supervised approach that can arbitrarily generate many images of words with arbitrary length from a generator \mathcal{G} to augment handwriting data and uses a discriminator \mathcal{D} and recognizer \mathcal{R} . The paper proposes results for original data with random affine augmentation using synthetic images and refinement.

2.2 Online Handwriting Recognition

Motion-based handwriting [20] and air-writing [122] from sensor-enhanced devices have been extensively investigated. While such motions are spacious, the hand and pen motions for writing on paper are comparatively small-scale [15]. Research for classifying text from sensor-enhanced pens has recently attracted substantial interest. [45] use acceleration and audio data of handwritten actions for character recognition. Furthermore, recent publications came up with similar developments that are only prototypical, for example, [2, 47, 95]. Hence, there is already a lot of interest and future technical advancements will further boost this. The novel sensor-enhanced pen based on IMUs [77] enables new applications for writing on paper. Note that this pen is a finished product and can be bought. Data collection and processing is straightforward and allows applications to be easy to implement in real-world. [77] published the OnHW-chars dataset containing single characters. [56] evaluated the aleatoric and epistemic uncertainty to show the domain shift between right- and left-handed writers. [73] reduced this domain shift by adapting feature embeddings based on transformations from optimal transport techniques. [57] presented an approach for distributing the computational workload between a sensor pen and a mobile device (i.e., smartphone or tablet) for

handwriting recognition, as interference on mobile devices leads to high system requirements. [75] reconstructed the trajectory of the pen tip for single characters written on tablets from IMU data and cameras pointing at the pen tip [74]. A more challenging task than single-character classification is the classification of sequences (i.e., words or equations). [76] proposed several sequence-based datasets and a large benchmark of convolutional, recurrent, and Transformer-based architectures, loss functions, and augmentation techniques. While [111] combined a binary random forest to classify the writing activity and a CNN for windows of single-label predictions, [14] highlighted the effectiveness of Transformers for classifying equations. Methods such as [96] cannot be applied to this online task, as these methods are designed for image-based (offline) HWR, and traditional methods such as [16] for online HWR are based on online trajectories written on tablets. Recently, [6] evaluated further machine and deep learning models as well as deep ensembles on the single OnHW-chars dataset.

2.3 Cross-Modal Representation Learning

For traditional methods that learn a cross-modal representation, a cross-modal similarity for the retrieval can be calculated with linear projections [87]. However, cross-modal correlation is highly complex, and hence, recent methods are based on a *modal-sharing network* to jointly transfer non-linear knowledge from a single modality to all modalities [112]. [49] use a *cross-modal network* between different modalities (image to video, text, audio and 3D models) and a *single-modal network* (shared features between images of source and target domains). They use two convolutional layers (similar to our proposed architecture) that allow the model to adapt by using more trainable parameters. However, while their auxiliary network uses the same modality, the auxiliary network of the proposed method in this paper is based on another modality. [59] learn a cross-modal embedding between video frames and audio signals with graph clusters, but both modalities must be available at inference. [88] proposed an image-text modality adversarial matching approach that learns modality-invariant feature representations, but their projection loss is only used for learning discriminative image-text embeddings. [42] propose a model for single-modal inference. However, they use image and depth modalities for person re-identification without a time-series component, which makes the problem considerably different. [63] handled multi-sensory modalities for 3D models only. For an overview of CMR, see [24]. An overview of relevant CMR methods is given in Table 8 in the Appendix A.3. With respect to the kind of the modality, the work by [40, 73] is closest, while the applications in [73, 106, 120] of handwriting recognition are relevant.

2.4 Pairwise and Triplet Learning

Networks trained for a classification task can produce useful feature embeddings with efficient runtime complexity $O(NC)$ per epoch, where N is the number of training samples and C is the number of classes. However, the classical cross-entropy (CE) loss is only partly useful for deep metric learning, as it ignores how close each point is to its class centroid (or how far apart each point is from other class centroids). CE variations (e.g., for face recognition) that learn angularly discriminative features have also been developed [66]. The *pairwise contrastive loss* [22] minimizes the distance between feature embedding pairs of the same class and maximizes the distance between feature embedding pairs of different classes depending on a margin parameter. The drawback is that the optimization of positive pairs is independent of negative pairs, but the optimization should force the distance between positive pairs to be smaller than negative pairs [29].

The *triplet loss* [116] addresses this by defining an anchor and a positive point as well as a negative point and forces the positive pair distance to be smaller than the negative pair distance by a certain margin. The runtime complexity of the triplet loss is $O(N^3/C)$ and can be computationally challenging for large training sets. Hence, several approaches exist to reduce this complexity, such as hard or semi-hard triplet mining [90] and smart triplet mining [44]. Often, data evolve over time, and

hence, [91] proposed a formulation of the triplet loss where the traditional static *margin* is superseded by a temporally adaptive maximum margin function. While [61, 120] combine the triplet loss with the CE loss, [41] use a triplet selection with L_2 -normalization for language modeling, but considered all negative pairs for triplet selection with fixed similarity intensity parameter. The proposed method uses a triplet loss with a dynamic margin together with a novel word-level triplet selection. The TNN-C-CCA [119] also uses the triplet loss on embeddings between an anchor from audio data and positive and negative samples from visual data and the cosine similarity for the final representation comparison. In image-to-caption CMR tasks, the most common design is separated encoders that allow the separated inference without the other modality [19, 32]. We choose a similar separate cross-modal encoder for single-modal inference. CrossATNet [18], another triplet loss-based method that uses single class labels, defines class sketch instances as the anchor, the same class image instance as the positive sample, and a different class image instance as the negative sample. While the previous methods are based on a triplet selection method using single-label classification, related work exists for using the triplet loss for sequence-based classification (i.e., from texts) [12, 26, 36, 123]. To the best of our knowledge, no approach so far has used triplet-based cross-modal learning based on the Edit distance between words. Most relevant are the works by [18, 19, 42, 72, 107] that use the triplet loss, but without a dynamic margin.

2.5 Deep Metric Learning

As deep metric learning is a very broad and advanced field, only the most related work is described here. For an overview of deep metric learning, see [71]. Most of the related work uses the Euclidean metric as distance loss, although the triplet loss can be defined based on any other (sub-)differentiable distance metric. [106] proposed a method for offline signature verification based on a dual triplet loss that uses the Euclidean space to project an input image to an embedding function. While [86] use the Euclidean metric to learn the distance between feature embeddings, [120] use the Cosine similarity. [48] state that using the *non-squared* Euclidean distance is more stable, while the *squared* distance made the optimization more prone to collapsing. Recent methods extend the canonical correlation analysis (CCA) [85] that learns linear projection matrices by maximizing pairwise correlation of cross-modal data. To share information between the same modality (i.e., images), the maximum mean discrepancy (MMD) [68] is typically minimized.

3 METHODOLOGICAL BACKGROUND

We define the problem of cross-modal representation learning and present deep metric learning loss functions in Section 3.1. In Section 3.2, we propose the triplet loss for cross-modal learning.

3.1 Cross-Modal Retrieval for Time-Series and Image Classification

A multivariate time-series $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$ is an ordered sequence of $l \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l})$, $i \in \{1, \dots, m\}$, where $m \in \mathbb{N}$ is the length of the time-series. The multivariate time-series training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\} \in \mathbb{R}^{n_U \times m \times l}$, where n_U is the number of time-series. Let $\mathbf{X} \in \mathbb{R}^{h \times w}$ with entries $x_{i,j} \in [0, 255]$ represent an image from the image training set. The image training set is a subset of the array $\mathcal{X} = \{\mathbf{X}_1, \dots, \mathbf{X}_{n_X}\} \in \mathbb{R}^{n_X \times h \times w}$, where n_X is the number of time-series. The aim of joint multivariate time-series and image classification tasks is to predict an unknown class label $y \in \Omega$ for single class prediction or $\mathbf{y} \in \Omega$ for sequence prediction for a given multivariate time-series or image (see also Section 4.2). The time-series samples denote the main training data, while the image samples represent the privileged information that is not used for inference. In addition to good prediction performance, the goal is to learn representative embeddings $f_c(\mathbf{U})$ and $f_c(\mathbf{X}) \in \mathbb{R}^{q \times t}$ to map multivariate time-series and image data into a feature

space $\mathbb{R}^{q \times t}$, where f_c is the output of the convolutional layer(s) $c \in \mathbb{N}$ of the latent representation and $q \times t$ is the dimension of the layer output.

We force the embedding to live on the $q \times t$ -dimensional hypersphere by using softmax – i.e., $\|f_c(\mathbf{U})\|_2 = 1$ and $\|f_c(\mathbf{X})\|_2 = 1 \forall c$ (see [113]). In order to obtain a small distance between the embeddings $f_c(\mathbf{U})$ and $f_c(\mathbf{X})$, we minimize deep metric learning functions $\mathcal{L}_{\text{DML}}(f_c(\mathbf{X}), f_c(\mathbf{U}))$. Well-known deep learning metric are the distance-based mean squared error (MSE) \mathcal{L}_{MSE} , the spatio-temporal cosine similarity (CS) \mathcal{L}_{CS} , the Pearson correlation (PC) \mathcal{L}_{PC} , and the distribution-based Kullback-Leibler (KL) divergence \mathcal{L}_{KL} . In our experiments, we additionally evaluate the kernelized maximum mean discrepancy (kMMD) $\mathcal{L}_{\text{kMMD}}$, Bray Curtis (BC) \mathcal{L}_{BC} , and Poisson \mathcal{L}_{PO} losses. We study their performance in Section 5. A combination of classification and cross-modal representation learning losses can be realized by dynamic weight averaging [65] as a multi-task learning approach that performs dynamic task weighting over time (see Appendix A.4).

3.2 Contrastive Learning and Triplet Loss

While the training with the previous loss functions uses inputs where the image and multivariate time-series have the same label, pairs with similar but different labels can improve the training process. This can be achieved using the triplet loss [90], which enforces a margin between pairs of image and multivariate time-series data with the same identity to all other different identities. As a consequence, the convolutional output for one and the same label lives on a manifold, while still enforcing the distance – and thus, discriminability – to other identities.

Therefore, we seek to ensure that the embedding of the multivariate time-series \mathbf{U}_i^a (*anchor*) of a specific label is closer to the embedding of the image \mathbf{X}_i^p (*positive*) of the same label than it is to the embedding of any image \mathbf{X}_i^n (*negative*) of another label (see Figure 2).

Thus, we want the following inequality to hold for all training samples $(f_c(\mathbf{U}_i^a), f_c(\mathbf{X}_i^p), f_c(\mathbf{X}_i^n)) \in \Phi$:

$$\mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{X}_i^p)) + \alpha < \mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{X}_i^n)), \quad (1)$$

where $\mathcal{L}_{\text{DML}}(f_c(\mathbf{X}), f_c(\mathbf{U}))$ is a deep metric learning loss, α is a margin between positive and negative pairs, and Φ is the set of all possible triplets in the training set. The *contrastive loss* minimizes the distance of the anchor to the positive sample and separately maximizes the distance to the negative sample. Instead, based on (1), we can formulate a differentiable loss function – the *triplet loss* – that we can use for optimization:

$$\mathcal{L}_{\text{trpl,c}}(\mathbf{U}^a, \mathbf{X}^p, \mathbf{X}^n) = \sum_{i=1}^N \max \left[\mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{X}_i^p)) - \mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{X}_i^n)) + \alpha, 0 \right], \quad (2)$$

where $c \in \mathbb{N}$.² Selecting negative samples that are too close to the anchor (in relation to the positive sample) can cause slow training convergence. Hence, triplet selection must be handled carefully and with consideration for each specific application [29]. We choose negative samples based on the class distance (single labels) and on the Edit distance (sequence labels) (see Section 4.2).

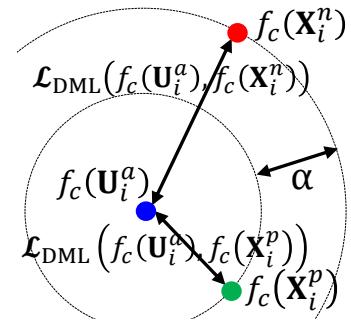


Fig. 2. Triplet pair.

²To have a larger number of trainable parameters in the latent representation with a greater depth, we evaluate one and two stacked convolutional layers, each trained with a shared loss $\mathcal{L}_{\text{trpl,c}}$.

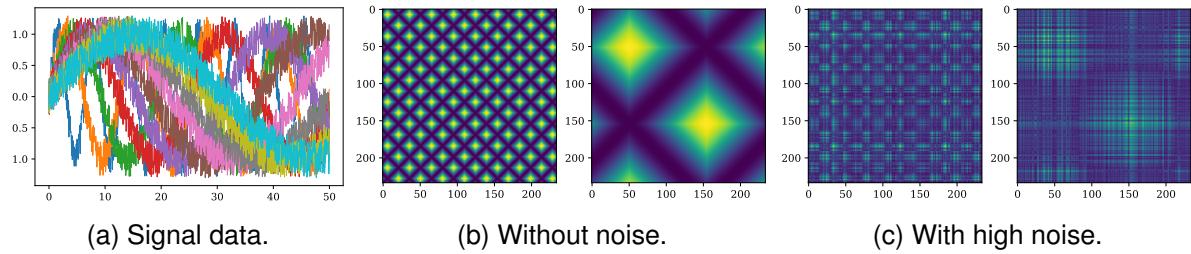


Fig. 3. Synthetic signal data (a) for 10 classes, and image data (b-c) for classes 0 (left) and 6 (right).

4 METHOD

We now demonstrate the efficacy of our proposal. In Section 4.1, we generate sinusoidal time-series with introduced noise (main task) and compute the corresponding Gramian angular summation field with different noise parameters (auxiliary task) (see Figure 1). In Section 4.2, we combine online (inertial sensor signals, main task) and offline data (visual representations, auxiliary task) for HWR with sensor-enhanced pens. This task is particularly challenging, due to different data representations based on images and multivariate time-series data. For both applications, our approach allows to only use the main modality (i.e., multivariate time-series) for inference. We further analyze and evaluate different deep metric learning functions to minimize the distance between the learned embeddings.

4.1 Cross-Modal Learning on Synthetic Data

We first investigate the influence of the triplet loss for cross-modal learning between synthetic time-series and image-based data as a sanity check. For this, we generate signal data of 1,000 timesteps with different frequencies for 10 classes (see Figure 3a) and add noise from a continuous uniform distribution $U(a, b)$ for $a = 0$ and $b = 0.3$. We use a recurrent CNN with the CE loss to classify these signals. From each signal without noise, we generate a Gramian angular summation field [110]. For classes with high frequencies, this results in a fine-grained pattern, and for low frequencies in a coarse-grained pattern. We generate Gramian angular summation fields with different added noise between $b = 0$ (Figure 3b) and $b = 1.95$ (Figure 3c). A small CNN classifies these images with the CE loss. To combine both networks, we train each signal-image pair with the triplet loss. As the frequency of the sinusoidal signal is closer for more similar class labels, the distance in the manifold embedding should also be closer. For each batch, we select negative sample pairs for samples with the class label $CL = 1 + \lfloor \frac{\max_e - e - 1}{25} \rfloor$ as the lower bound for the current epoch e and the maximum epoch \max_e . We set the margin α in the triplet loss separately for each batch such that $\alpha = \beta \cdot (CL_p - CL_n)$ depends on the positive CL_p and negative CL_n class labels of the batch and is in the range [1, 5] with $\beta = 0.1$. The batch size is 100 and $\max_e = 100$. Appendix A.5 provides further details. This combination of the CE loss with the triplet loss can lead to a mutual improvement of the utilization of the classification task and embedding learning.

4.2 Cross-Modal Learning for HWR

Method Overview. Figure 4 gives a method overview. The main task is online HWR to classify words written with a sensor-enhanced pen and represented by multivariate time-series of the different pen sensors. To improve the classification task with a better generalizability, the auxiliary network performs offline HWR based on an image input. We pre-train ScrabbleGAN [35] on the IAM-OffDB [67] dataset. For all time-series word labels, we then generate the corresponding image as the positive time-series-image pair. Each multivariate time-series and each image is associated with y – a sequence of L class labels from a pre-defined label set Ω with K classes. For our classification

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss

232

Functions for Online Handwriting Recognition

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions

000:9

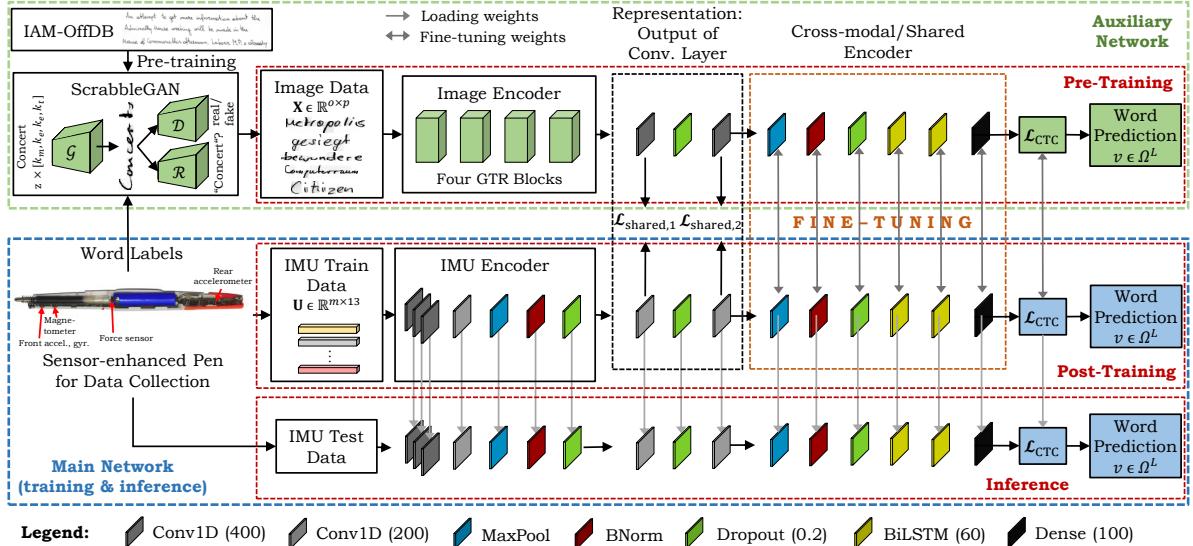


Fig. 4. Detailed method overview: The middle pipeline consists of data recording with a sensor-enhanced pen, feature extraction of inertial multivariate time-series data, and word classification with CTC. We generate image data with the pre-trained ScrabbleGAN for corresponding word labels. The top pipeline (four gated text recognizer blocks) extracts features from images. The distances of the embeddings are minimized with the triplet loss and deep metric learning functions. The classification network with two BiLSTM layers are fine-tuned for the OnHW task for a cross-modal representation.

task, $y \in \Omega^L$ describes words. The multivariate time-series training set is a subset of the array \mathcal{U} with labels $\mathcal{Y}_U = \{y_1, \dots, y_{n_U}\} \in \Omega^{n_U \times L}$. The image training set is a subset of the array X , and the corresponding labels are $\mathcal{Y}_X = \{y_1, \dots, y_{n_X}\} \in \Omega^{n_X \times L}$. Offline HWR techniques are based on Inception, ResNet34, or gated text recognizer [118] modules. The architecture of the online HWR method consists of an IMU encoder with three 1D convolutional layers of size 400, a convolutional layer of size 200, a max pooling and batch normalization, and a dropout of 20%. The online method is improved by sharing layers with a common representation by minimizing the distance of the feature embedding of the convolutional layers $c \in \{1, 2\}$ (integrated in both networks) with a shared loss $\mathcal{L}_{\text{shared},c}$. We set the embedding size $\mathbb{R}^{q \times t}$ to 400×200 . Both networks are trained with the connectionist temporal classification (CTC) [38] loss \mathcal{L}_{CTC} to avoid pre-segmentation of the training samples by transforming the network outputs into a conditional probability distribution over label sequences.

Datasets for Online HWR. We make use of two word datasets proposed in [76]. These datasets are recorded with a sensor-enhanced pen that uses two accelerometers (3 axes each), one gyroscope (3 axes), one magnetometer (3 axes), and one force sensor at 100 Hz [75, 77]. One sample of size $m \times l$ represents an multivariate time-series of a written word of m timesteps from $l = 13$ sensor channels. One word is a sequence of small or capital characters (52 classes) or with mutated vowels (59 classes). The *OnHW-words500* dataset contains 25,218 samples where each of the 53 writers contributed the same 500 words. The *OnHW-wordsRandom* dataset contains 14,641 randomly selected words from 54 writers. For both datasets, 80/20 train/validation splits are available for writer-(in)dependent (WD/WI) tasks. We transform (zero padding, interpolation) all samples to 800 timesteps. For more information on the datasets, see [76].

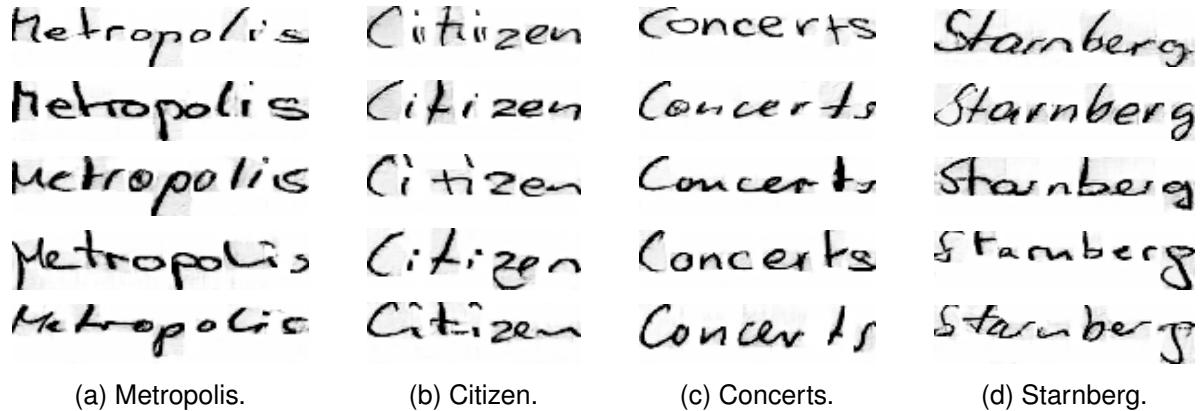


Fig. 5. Overview of four generated words with ScrabbleGAN [35] with various text styles.

Image Generation for Offline HWR. In order to couple the online time-series data with offline image data, we use a generative adversarial network (GAN) to arbitrarily generate many images.

ScrabbleGAN [35] is a state-of-the-art semi-supervised approach that consists of a generator \mathcal{G} that generates images of words with arbitrary length from an input word label, a discriminator \mathcal{D} , and a recognizer \mathcal{R} that promotes style and data fidelity. While \mathcal{D} promotes realistic-looking handwriting styles, \mathcal{R} encourages the result to be readable. ScrabbleGAN minimizes a joint loss term $\mathcal{L} = \mathcal{L}_D + \lambda \mathcal{L}_R$ where \mathcal{L}_D and \mathcal{L}_R are the loss terms of \mathcal{D} and \mathcal{R} , respectively, and the balance factor is λ . The generator \mathcal{G} is designed such that each character is generated individually, using the property of the convolutions of overlapping receptive fields to account for the influence of nearby letters. Four character filters (k_m, k_e, k_e and k_t) are concatenated, multiplied by a noise vector z , and fed into a class-conditioned generator (see Figure 6). This allows for adjacent characters to interact and creates a smooth transition, e.g., enabling cursive text. The style of the image is controlled by a noise vector z given as the input to the network (being consistent for all characters of a word). The recognizer \mathcal{R} discriminates between real and gibberish text by comparing the output of \mathcal{R} to the one that was given as input to \mathcal{G} . \mathcal{R} is trained only on real and labeled samples. \mathcal{R} is inspired by CRNN [94] and uses the CTC [38] loss. The architecture of the discriminator \mathcal{D} is inspired by BigGAN [13] consisting of four residual blocks and a linear layer with one output. \mathcal{D} is fully convolutional, predicts the average of the patches, and is trained with a hinge loss [64]. We train ScrabbleGAN with the IAM-OffDB [67] dataset and generate three different datasets. Exemplary images are shown in Figure 5. First, we generate 2 million images randomly selected from a large lexicon (*OffHW-German*), and pre-train the offline HWR architectures. Second, we generate 100,000 images based on the same word labels for each of the *OnHW-words500* and *OnHW-wordsRandom* datasets (*OffHW-words500*, *OffHW-wordsRandom*) and fine-tune the offline HWR architectures.

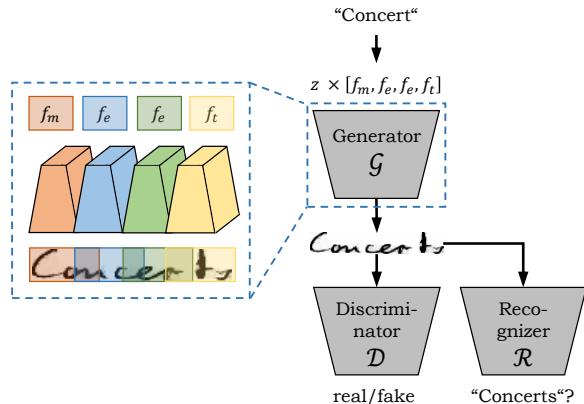
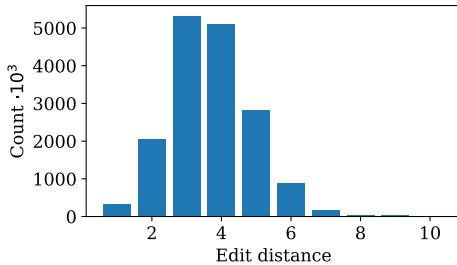
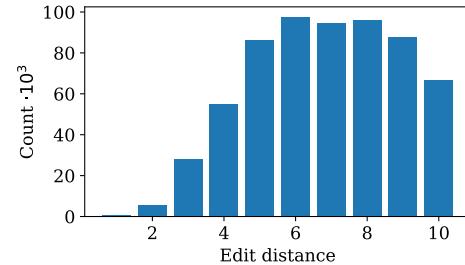


Fig. 6. ScrabbleGAN concept by [35] of generating the word "Concerts".



(a) OnHW-words500.



(b) OnHW-wordsRandom.

Fig. 7. Number image-time-series pairs dependent on substitutions.

Methods for Offline HWR. OrigamiNet [117] is a state-of-the-art multi-line recognition method using only unsegmented image and text pairs. Similar to OrigamiNet, our offline method is based on different encoder architectures with one or two additional 1D convolutional layers (each with filter size 200, softmax activation [120]) with 20% dropout for the latent representation, and a cross-modal representation decoder with BiLSTMs. For the encoder, we make use of Inception modules from GoogLeNet [99] and the ResNet34 [46] architectures, and we re-implement the newly proposed gated, fully-convolutional method termed the gated text recognizer [118]. See Appendix A.6 for detailed information on the architectures. We train the networks on the generated OffHW-German dataset for 10 epochs and fine-tune on the OffHW-[500, wordsRandom] datasets for 15 epochs. For comparison with state-of-the-art techniques, we train OrigamiNet and compare with IAM-OffDB. For OrigamiNet, we apply interline spacing reduction via seam carving [5], resizing the images to 50% height, and random projective (rotating and resizing lines) and random elastic transform [114]. We augment the OffHW-German dataset with random width resizing and apply no augmentation for the OffHW-[words500, wordsRandom] datasets for fine-tuning.

Offline/Online Cross-Modal Representation Learning. Our architecture for online HWR is based on [76]. The encoder extracts features of the inertial data and consists of three convolutional layers (each with filter size 400, ReLU activation) and one convolutional layer (filter size 200, ReLU activation), a max pooling, batch normalization and a 20% dropout layer. As for the offline architecture, the network then learns a latent representation with one or two convolutional layers (each with filter size 200, softmax activation) with 20% dropout and the same cross-modal representation decoder. The output of the convolutional layers of the latent representation are minimized with the $\mathcal{L}_{\text{shared},c}$ loss. The layers of the common representation are fine-tuned based on the pre-trained weights of the offline technique. Here, two BiLSTM layers with 60 units each and ReLU activation extract the temporal context of the feature embedding. As for the baseline classifier, we train for 1,000 epochs. For evaluation, the main time-series network is independent of the image auxiliary network by using only the weights of the main network.

Triplet Selection. To ensure (fast) convergence, it is crucial to select triplets that violate the constraint from Equation 1. Typically, it is infeasible to compute the loss for all triplet pairs, or this leads to poor training performance (as poorly chosen pairs dominate hard ones). This requires an elaborate triplet selection [29]. We use the Edit distance to define the identity and select triplets. The Edit distance is the minimum number of substitutions S , insertions I , and deletions D required to change the sequences $\mathbf{d} = (d_1, \dots, d_r)$ into $\mathbf{g} = (g_1, \dots, g_z)$ with length r and z , respectively. We define two sequences with an Edit distance of 0 as the positive pair, and with an Edit distance larger than 0 as the negative pair. Based on preliminary experiments, we use only substitutions for triplet

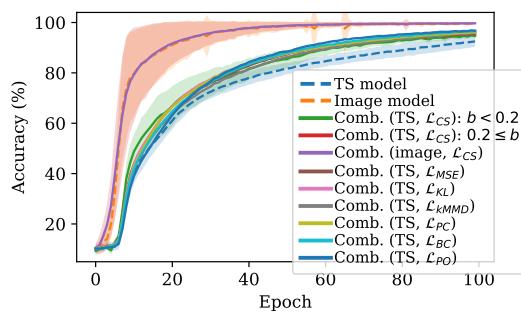
selection that lead to a higher accuracy compared to additional insertions and deletions (whereas these would also change the length difference of image and time-series pairs). We constrain $p - m/2$ (the difference in pixels p of the images and half the number of timesteps of the time-series) to be maximally ± 20 . The goal is to achieve a small distance for positive pairs and a large distance for negative pairs that increases with a larger Edit distance (between 1 and 10). Furthermore, despite a limited number of word labels, there still exist a large number of image-time-series pairs per word label for every possible Edit distance (see Figure 7). For each batch, we search in a dictionary of negative sample pairs for samples with $Edit_distance = 1 + \lfloor \frac{\max_e - e - 1}{100} \rfloor$ as the lower bound for the current epoch e and maximal epochs \max_e . For every label, we randomly pick one image. We let the margin α in the triplet loss vary for each batch such that $\alpha = \beta \cdot \overline{Edit_distance}$ depends on the mean Edit distance of the batch and is in the range $[1, 11]$ with $\beta = 10^{-3}$ for MSE, $\beta = 0.1$ for CS and PC, and $\beta = 1$ for KL. The batch size is 100 and $\max_e = 1,000$.

5 EXPERIMENTAL RESULTS

Hardware and Training Setup. For all experiments, we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the vanilla Adam optimizer with a learning rate of 10^{-4} .

5.1 Evaluation of Synthetic Data

We train the time-series (TS) model 18 times with noise $b = 0.3$ and the combined model with the triplet loss for all 40 noise combinations ($b \in \{0, \dots, 1.95\}$) with different deep metric learning functions. Figure 8 shows the validation accuracy averaged over all trainings as well as the combined cases separately for noise $b < 0.2$ and noise $0.2 \leq b < 2.0$ (for the \mathcal{L}_{CS} loss). Table 1 summarizes the final classification results of all cases. The accuracy of the models that use only images and in combination with time-series during inference reach an accuracy of 99.7% (which can be seen as an unreachable upper bound for the TS-only models). The triplet loss improves the final TS baseline accuracy from 92.5% to 95.36% (averaged over all combinations), while combining TS and image data leads to a faster convergence. Conceptually similar to [68], we use the \mathcal{L}_{kMMD} loss, which yields 95.83% accuracy. The \mathcal{L}_{PC} (96.03%), \mathcal{L}_{KL} (96.22%), \mathcal{L}_{MSE} (96.25%), \mathcal{L}_{BC} (96.62%), and \mathcal{L}_{PO} (96.76%) loss functions can further improve the accuracy. We conclude that the triplet loss can be successfully used for cross-modal learning by utilizing negative identities.



Method	Accuracy (%)
TS model	92.50
Combined (TS, \mathcal{L}_{CS})	95.36
Combined (image, \mathcal{L}_{CS})	99.70
Combined (TS, \mathcal{L}_{MSE})	96.25
Combined (TS, \mathcal{L}_{KL})	96.22
Combined (TS, \mathcal{L}_{kMMD})	95.83
Combined (TS, \mathcal{L}_{PC})	96.03
Combined (TS, \mathcal{L}_{BC})	96.62
Combined (TS, \mathcal{L}_{PO})	96.76

Table 1. Comparison of single- and cross-modal representation learning.

Fig. 8. Accuracy of single- and cross-modal representation learning over all epochs.

5.2 Evaluation of Handwriting Recognition

Evaluation Metrics. A metric for sequence evaluation is the character error rate (CER), defined as $CER = \frac{S_c + I_c + D_c}{N_c}$, i.e., the Edit distance (the sum of character substitutions S_c , insertions I_c and deletions D_c) divided by the total number of characters in the set N_c . Similarly, the word error rate

Table 2. Evaluation results (WER and CER in %) for the generated dataset with ScrabbleGAN [35] OffHW-German and the IAM-OffDB [67] dataset.

	Method	OffHW-German		IAM-OffDB	
		WER	CER	WER	CER
Related Work	ScrabbleGAN [35] OrigamiNet [117] (12 × gated text recognizer)	-	-	23.61	-
		-	-	-	4.70
Our Implementation	OrigamiNet (ours, 4 × gated text recognizer)	1.50	0.11	90.40	15.67
	Inception	12.54	1.17	-	-
	ResNet	13.05	1.24	-	-
	Gated text recognizer (2 blocks), 1 conv. layer	4.34	0.39	-	-
	Gated text recognizer (2 blocks), 2 conv. layer	5.02	0.44	-	-
	Gated text recognizer (4 blocks), 1 conv. layer	3.35	0.34	89.37	15.60
	Gated text recognizer (4 blocks), 2 conv. layer	2.52	0.24	-	-
	Gated text recognizer (6 blocks)	2.85	0.26	-	-
	Gated text recognizer (8 blocks)	4.22	0.38	-	-

Table 3. Evaluation results (WER and CER in %) for the generated OffHW-words500 and OffHW-wordsRandom datasets for one and two convolutional layers (c). We propose writer-dependent (WD) and writer-independent (WI) results.

Method (4 × gated text recognizer)	OffHW-words500				OffHW-wordsRandom			
	WD		WI		WD		WI	
	WER	CER	WER	CER	WER	CER	WER	CER
$c = 1$	2.94	0.76	0.95	0.23	1.98	0.35	2.05	0.37
$c = 2$	2.51	0.69	0.85	0.22	1.82	0.34	1.95	0.38

(WER) is defined as $WER = \frac{S_w + I_w + D_w}{N_w}$, which is computed with the sum of word operations S_w , I_w and D_w , divided by the number of words in the set N_w .

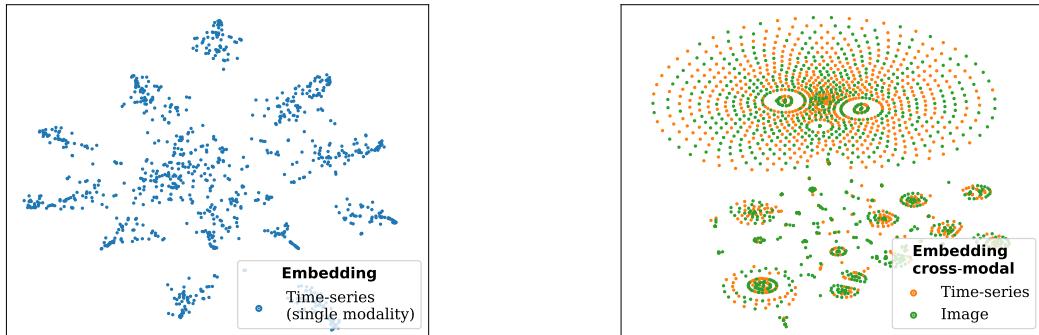
Evaluation of Offline HWR Methods. Table 2 shows offline HWR results on our generated OffHW-German dataset and on the IAM-OffDB [67] dataset. ScrabbleGAN [35] yields a WER of 23.61% on the IAM-OffDB dataset, while OrigamiNet [117] achieves a CER of 4.70% with 12 gated text recognizer modules. While OrigamiNet is trained for the multi-line classification, which is an easier task (as the image of the paragraph does not have to be segmented into lines), we trained OrigamiNet on single-lines with zero padding, which is closer to the OffHW-German dataset. While the images for the multi-line task are of approximately similar lengths, the image lengths of the single-line task varies strongly, and hence, zero padding has a high influence on the model performance, resulting in a CER of 15.67%. While [117] did not propose WER results, OrigamiNet yields only a WER of 90.40%. This problem does not appear for the OffHW-German dataset, as the dataset contains only single words with similar lengths. With our own implementation of four gated text recognizer modules and one convolutional layer for the common representation, our model achieves similar results. As the training takes more than one day for one epoch on the large OffHW-German dataset, we train OrigamiNet with four gated text recognizer modules, and achieve 0.11% CER on the generated dataset and 15.67% on the IAM-OffDB dataset. All our models yield low error rates on the generated OffHW-German dataset. Our approach with gated text recognizer blocks outperforms (0.24% to 0.44% CER) the models with Inception [99] (1.17% CER) and ResNet [46] (1.24% CER). OrigamiNet achieves the lowest error rates of 1.50% WER and 0.11% CER. Four gated text recognizer blocks yield the best results at a significantly lower training time compared to six or eight

Table 4. Feature embeddings $f_c(\mathbf{X}_i)$ and $f_c(\mathbf{U}_i)$ of exemplary image \mathbf{X}_i and multivariate time-series \mathbf{U}_i data of the convolutional layer $c = \text{conv}_2$ for different deep metric learning functions for positive pairs ($\text{Edit_distance} = 0$) and negative pairs ($\text{Edit_distance} > 0$) trained with the triplet loss. The feature embeddings are similar in the red box (character x) or blue box (character p) for $f_2(\mathbf{X}_i)$, or the last pixels (character t) of $f_2(\mathbf{U}_i)$ for \mathcal{L}_{PC} marked green.

ED	Label	Image \mathbf{U}_i	$f_2(\mathbf{X}_i)$	$f_2(\mathbf{U}_i): \mathcal{L}_{\text{MSE}}$	$f_2(\mathbf{U}_i): \mathcal{L}_{\text{CS}}$	$f_2(\mathbf{U}_i): \mathcal{L}_{\text{PC}}$	$f_2(\mathbf{U}_i): \mathcal{L}_{\text{KL}}$
0	Export	<i>Export</i>					
1	Expert	<i>Expert</i>					
2	Import	<i>Import</i>					
3	Vorort	<i>Vorort</i>					

blocks. We fine-tune the model with four gated text recognizer blocks for one and two convolutional layers and achieve notably low error rates between 0.22% to 0.76% CER, and between 0.85% to 2.95% WER on the OffHW-[words500, wordsRandom] datasets (see Table 3). While results for OffHW-wordsRandom are similar for writer-dependent (WD) and writer-independent (WI) tasks, WI results of the OffHW-words500 dataset are lower than WD results, as words with the same label appear in the training and test dataset. We use the weights of the fine-tuning as initial weights of the image model for the cross-modal representation learning.

Evaluation of Representation Learning Feature Embeddings. Table 4 shows the feature embeddings for image $f_2(\mathbf{X}_i)$ and time-series data $f_2(\mathbf{U}_i)$ of the *positive* sample **Export** and the two *negative* samples **Expert** ($\text{Edit_distance} = 1$) and **Import** ($\text{Edit_distance} = 2$) based on four deep metric learning loss functions. The pattern of characters are similar, as the words differ only in the fourth letter. In contrast, **Import** has a different feature embedding, as the replacement of E with I and x with m leads to a higher feature distance in the embedding hypersphere. Note that image and time-series data can vary in length for $\text{Edit_distance} > 0$. Figure 9 shows the feature embeddings of the output of the convolutional layers ($c = 1$) processed with t-SNE [101]. Figure 9a visualizes the multivariate time-series embeddings $f_1(\mathbf{U}_i)$ of the single modal network. The learned representation generalizes well, but misclassifications (e.g., of small and capital letters at the beginning of a word, which happen quite often) also introduce errors in the latent representation. Figure 9b visualizes the multivariate time-series and image embeddings ($f_1(\mathbf{U}_i)$ and $f_1(\mathbf{X}_i)$, respectively) in a cross-modal setup. While the embedding of the single modal network is unstructured, the embeddings of the cross-modal network are structured (distance of samples visualizes the Edit distance between words) with the embeddings of the time-series modality being close to the embeddings of the image modality, and hence, more distinctive clusters with better separation.



(a) Feature embedding of IMU samples for the single modality network.

(b) Feature embeddings of IMU and image samples for the cross-modal network.

Fig. 9. Comparison of the naive method (left) and our proposed approach (right), where our method shows a much better behaved embedding space compared to the naive approach by learning a joint representation. Plot of 400×200 feature embeddings of image and IMU modalities with t-SNE.

Evaluation of Cross-Modal Representation Learning. Table 5 gives an overview of cross-modal representation learning (for $c = 1$). The first row shows baseline results by [76]: 13.04% CER on OnHW-words500 (WD) and 6.75% CER on OnHW-wordsRandom (WD) with mutated vowels. Compared to various time-series classification techniques, their benchmark results showed superior performance of CNN+BiLSTMs on these OnHW recognition tasks. Only InceptionTime [34] (a large time-series encoder network with $depth = 11$ and $nf = 96$) – with BiLSTM layers – yields partly better results or is on par with the CNN+BiLSTM model for sequence-based classification, while the CNN+BiLSTM model outperforms state-of-the-art techniques on single character-based classification tasks. Due to the faster training of the CNN+BiLSTM, we chose this network for the cross-modal task. In general, the word error rate (WER) can vary for a similar character error rate (CER). The reason is that a change of one character of a correctly classified word leads to a large change in the WER, while the change of the CER is marginal. We define the results trained without mutated vowels as baseline results, as ScrabbleGAN is pretrained on IAM-OffDB, which does not contain mutated vowels, and hence, such words cannot be generated. Nevertheless, the main model can be trained and is applicable to samples with mutated vowels. For a fair comparison, we compare our results to the results of the models trained without mutated vowels. Here, the error rates are slightly higher for both datasets. As expected, cross-modal learning improves the baseline results up to 11.28% CER on the OnHW-words500 WD dataset and up to 7.01% CER on the OnHW-wordsRandom WD dataset. The contrastive loss shows the best results on the OnHW-words500 (WD) dataset with the Kullback-Leibler metric and on the OnHW-wordsRandom dataset (WD) with the cosine similarity metric. With the triplet loss, \mathcal{L}_{CS} outperforms other metrics on the OnHW-wordsRandom dataset but is inconsistent on the OnHW-words500 dataset. The importance of the triplet loss is more significant for one convolutional layer ($c = 1$) than for two convolutional layers ($c = 2$) (see Appendix A.7). Furthermore, training with kMMD (implemented as in [68]) does not yield reasonable results. We assume that this metric cannot make use of the important time component in the HWR application. We proposed our approach as learning with privileged information by exploiting a visual modality as an auxiliary task and improve the main task based on an inertial modality. The cross-modal learning would also work for the visual modality as the main task and a generated dataset for the inertial modality as an auxiliary task. However, the error rates are already low for the image-based classification task, as methods for offline HWR are very

Table 5. Evaluation results (WER and CER in %) averaged over five splits of the baseline time-series-only technique and our cross-modal learning technique for the inertial-based OnHW datasets [76] with and without mutated vowels (MV) for one convolutional layer $c = 1$. Best results are **bold**, and second best results are underlined. Arrows indicate improvements (\uparrow) and degradation (\downarrow) of baseline results (w/o MV).

	Method	OnHW-words500				OnHW-wordsRandom			
		WD		WI		WD		WI	
		WER	CER	WER	CER	WER	CER	WER	CER
Main Task	InceptionTime, \mathcal{L}_{CTC} , w/ MV	37.12	12.96	62.09	26.36	42.88	7.19	84.14	32.35
	IT+BiLSTM, \mathcal{L}_{CTC} , w/ MV	43.22	13.07	61.62	26.08	39.14	6.39	85.42	33.31
	CNN+BiLSTM, \mathcal{L}_{CTC} , w/ MV	42.81	13.04	60.47	28.30	37.13	6.75	83.28	35.90
	CNN+BiLSTM, \mathcal{L}_{CTC} , w/o MV	42.77	13.44	59.82	28.54	38.02	7.81	83.54	36.51
Baseline	\mathcal{L}_{MSE}	40.76 \uparrow	12.71 \uparrow	55.54 \uparrow	<u>25.97</u> \uparrow	37.31 \uparrow	7.01 \uparrow	82.25 \uparrow	33.85 \uparrow
	\mathcal{L}_{CS}	38.62 \uparrow	11.55 \uparrow	<u>56.37</u> \uparrow	25.90 \uparrow	38.85 \downarrow	7.35 \uparrow	82.48 \uparrow	35.67 \uparrow
	\mathcal{L}_{PC}	39.09 \uparrow	11.69 \uparrow	57.90 \uparrow	27.23 \uparrow	38.46 \downarrow	7.15 \uparrow	82.71 \uparrow	35.13 \uparrow
	\mathcal{L}_{KL}	38.36 \uparrow	11.28 \uparrow	60.23 \downarrow	27.99 \uparrow	38.76 \downarrow	7.49 \uparrow	81.07 \uparrow	33.96 \uparrow
Contrastive Loss	$\mathcal{L}_{\text{contr},1}(\mathcal{L}_{MSE})$	38.34 \uparrow	11.57 \uparrow	56.81 \uparrow	<u>25.98</u> \uparrow	38.25 \downarrow	7.31 \uparrow	82.09 \uparrow	34.03 \uparrow
	$\mathcal{L}_{\text{contr},1}(\mathcal{L}_{CS})$	39.68 \uparrow	11.73 \uparrow	58.03 \uparrow	27.13 \uparrow	35.96 \uparrow	6.67 \uparrow	81.22 \uparrow	<u>33.11</u> \uparrow
	$\mathcal{L}_{\text{contr},1}(\mathcal{L}_{PC})$	37.82 \uparrow	11.34 \uparrow	57.45 \uparrow	26.18 \uparrow	39.22 \downarrow	7.39 \uparrow	82.45 \uparrow	34.21 \uparrow
	$\mathcal{L}_{\text{contr},1}(\mathcal{L}_{KL})$	36.70 \uparrow	10.84 \uparrow	61.72 \downarrow	29.16 \downarrow	38.92 \downarrow	7.51 \uparrow	83.54	35.52 \uparrow
Triplet Loss	$\mathcal{L}_{\text{trpl},1}(\mathcal{L}_{MSE})$	42.95 \downarrow	14.13 \downarrow	56.48 \uparrow	26.66 \uparrow	37.66 \uparrow	7.04 \uparrow	81.64 \uparrow	34.39 \uparrow
	$\mathcal{L}_{\text{trpl},1}(\mathcal{L}_{CS})$	38.01 \uparrow	11.29 \uparrow	58.50 \uparrow	27.10 \uparrow	<u>37.12</u> \uparrow	<u>6.98</u> \uparrow	82.71 \uparrow	33.09 \uparrow
	$\mathcal{L}_{\text{trpl},1}(\mathcal{L}_{PC})$	40.43 \uparrow	12.41 \uparrow	58.20 \uparrow	27.48 \uparrow	37.40 \uparrow	7.01 \uparrow	81.90 \uparrow	33.89 \uparrow
	$\mathcal{L}_{\text{trpl},1}(\mathcal{L}_{KL})$	<u>37.55</u> \uparrow	<u>11.21</u> \uparrow	63.52 \downarrow	30.52 \downarrow	38.39 \downarrow	7.36 \uparrow	83.18 \uparrow	35.21 \uparrow

advanced and the image dataset is very large. Hence, we assume that fine-tuning the image encoder with inertial data would result in a minor improvement. Prior work [76] evaluated data augmentation techniques for multivariate time-series data (i.e., time warping, scaling, jittering, magnitude warping, and shifting). This approach was rather limited with only 2-3% points of improvement compared with augmentation with the auxiliary image-based task.

Transfer Learning on Left-Handed Writers. To adapt the model to left-handed writers (who are typically under-represented and hence marginalized in the real-world), we make use of the left-handed datasets OnHW-words500-L and OnHW-wordsRandom-L proposed by [76]. These datasets contain recordings of two writers who provided 1,000 and 996 samples. As a baseline, we pre-train the time-series-only model on the right-handed datasets and post-train the left-handed datasets for 500 epochs (see the second and third rows of Table 6). As these datasets are rather small, the models can overfit on these specific writers and achieve a very low CER of 3.33% on the OnHW-words500-L datasets and 5.26% CER on the OnHW-wordsRandom-L dataset without mutated vowels for the writer-dependent tasks. However, the models cannot generalize on the writer-independent tasks, as evidenced by 62.07% CER on the OnHW-words500-L dataset and 81.15% CER on the OnHW-wordsRandom-L dataset. Hence, we focus on the WD tasks. For comparison, we use the state-of-the-art time-series classification technique InceptionTime [34] with $depth = 11$ and $nf = 96$ (without pre-training). As shown, our CNN+BiLSTM outperforms InceptionTime by a considerable margin. We use the weights of the pre-training with the offline handwriting datasets and again post-train on the left-handed datasets with $c = 1$ and $c = 2$. Using the weights of the cross-modal learning without the triplet loss can decrease the error rates up to 2.57% CER with \mathcal{L}_{KL} and 4.47% CER with \mathcal{L}_{PC} . Using the triplet loss $\mathcal{L}_{\text{trpl},2}(\mathcal{L}_{MSE})$ can further significantly decrease the WI OnHW-words500-L error

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss

240

Functions for Online Handwriting Recognition

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions

000:17

Table 6. Evaluation results (WER and CER in %) averaged over five splits of the baseline time-series-only technique and our cross-modal techniques for the inertial-based left-handed writers OnHW datasets [76] with and without mutated vowels (MV) for one ($c = 1$) and two ($c = 2$) convolutional layers $c = 1$. Best results are **bold**, and second best results are underlined. Arrows indicate improvements (\uparrow) and degradation (\downarrow) of baseline results (w/o MV).

	Method	OnHW-words500-L				OnHW-wordsRandom-L			
		WD		WI		WD		WI	
		WER	CER	WER	CER	WER	CER	WER	CER
Main Task	InceptionTime, \mathcal{L}_{CTC} , w/ MV	49.70	14.02	100.0	96.06	48.10	8.63	100.0	95.93
	CNN+BiLSTM, \mathcal{L}_{CTC} , w/ MV	14.20	3.30	94.40	71.41	30.20	4.86	100.0	83.51
	CNN+BiLSTM, \mathcal{L}_{CTC} , w/o MV	12.94	3.33	<u>89.07</u>	62.07	30.89	5.26	100.0	81.15
Baseline	\mathcal{L}_{MSE}	<u>11.62</u> \uparrow	2.77 \uparrow	90.65 \downarrow	67.90 \downarrow	30.53 \uparrow	4.93 \uparrow	100.0	81.99 \downarrow
	\mathcal{L}_{CS}	14.92 \downarrow	3.53 \downarrow	94.14 \downarrow	65.10 \downarrow	29.06 \uparrow	4.87 \uparrow	100.0	83.94 \downarrow
	\mathcal{L}_{PC}	12.29 \uparrow	3.04 \uparrow	91.33 \downarrow	60.89 \uparrow	<u>27.32</u> \uparrow	4.47 \uparrow	100.0	85.09 \downarrow
	\mathcal{L}_{KL}	11.37 \uparrow	2.57 \uparrow	93.02 \downarrow	66.64 \downarrow	29.61 \uparrow	4.91 \uparrow	100.0	81.28 \downarrow
Triplet Loss	$\mathcal{L}_{trpl,1}(\mathcal{L}_{MSE})$	12.48 \uparrow	3.11 \uparrow	90.09 \downarrow	62.87 \downarrow	32.62 \downarrow	5.43 \downarrow	100.0	80.41 \uparrow
	$\mathcal{L}_{trpl,1}(\mathcal{L}_{CS})$	13.65 \downarrow	3.28 \uparrow	90.76 \downarrow	62.40 \downarrow	34.21 \downarrow	5.53 \downarrow	100.0	82.14 \downarrow
	$\mathcal{L}_{trpl,1}(\mathcal{L}_{PC})$	13.71 \downarrow	3.23 \uparrow	91.55 \downarrow	65.95 \downarrow	31.59 \downarrow	5.32 \downarrow	100.0	81.77 \downarrow
	$\mathcal{L}_{trpl,1}(\mathcal{L}_{KL})$	13.65 \downarrow	3.45 \downarrow	94.93 \downarrow	72.01 \downarrow	31.87 \downarrow	5.42 \downarrow	100.0	82.02 \downarrow
	$\mathcal{L}_{trpl,2}(\mathcal{L}_{MSE})$	11.97 \uparrow	2.83 \uparrow	84.34 \uparrow	57.84 \uparrow	27.19 \uparrow	4.79 \uparrow	99.87 \uparrow	82.60 \downarrow
	$\mathcal{L}_{trpl,2}(\mathcal{L}_{CS})$	11.65 \uparrow	<u>2.63</u> \uparrow	94.70 \downarrow	67.69 \downarrow	28.39 \uparrow	<u>4.62</u> \uparrow	100.0	83.44 \downarrow
	$\mathcal{L}_{trpl,2}(\mathcal{L}_{PC})$	13.02 \downarrow	2.94 \uparrow	89.86 \downarrow	<u>60.26</u> \uparrow	30.22 \uparrow	4.81 \uparrow	100.0	84.29 \downarrow
	$\mathcal{L}_{trpl,2}(\mathcal{L}_{KL})$	13.55 \downarrow	3.22 \uparrow	97.86 \downarrow	76.54 \downarrow	28.14 \uparrow	4.71 \uparrow	100.0	<u>80.81</u> \uparrow

rates. In conclusion, due to the use of the weights of the cross-modal setup, the model can adapt faster to new writers and generalize better to unseen words due to the triplet loss.

6 CONCLUSION

We evaluated metric learning-based triplet loss functions for cross-modal representation learning between image and time-series modalities with class label-specific triplet selection. On synthetic data as well as on different HWR datasets, our method yields notable accuracy improvements for the main time-series classification task and can be decoupled from the auxiliary image classification task at inference time. Our cross-modal triplet selection further yields a faster training convergence with better generalization on the main task.

ACKNOWLEDGMENTS

This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A (David Rügamer) and by the research program Human-Computer-Interaction through the project “Schreibtrainer”, Grant No. 16SV8228, as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

REFERENCES

- [1] Emre Aksan, Fabrizio Pece, and Otmar Hilliges. 2018. DeepWriting: Making Digital Ink Editable via Deep Generative Modeling. In *CHI Conf. on Human Factors in Computing Systems*. 1–14. <https://doi.org/10.1145/3173574.3173779>
- [2] Tsige Tadesse Alemayoh, Masaaki Shintani, Jae Hoon Lee, and Shingo Okamoto. 2022. Deep-Learning-Based Character Recognition from Handwriting Motion Data Captured Using IMU and Force Sensors. In *MDPI Sensors*, Vol. 22(20). <https://doi.org/10.3390/s22207840>
- [3] Fevzi Alimoglu and Ethem Alpaydin. 1997. Combining Multiple Representations and Classifiers for Pen-based Handwritten Digit Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, Vol. 2. Ulm,

Germany. <https://doi.org/10.1109/ICDAR.1997.620583>

[4] Eloï Alonso, Bastien Moysset, and Ronaldo Messina. 2019. Adversarial Generation of Handwritten Text Images Conditioned on Sequences. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Sydney, Australia. <https://doi.org/10.1109/ICDAR.2019.00083>

[5] Shai Avidan and Ariel Shamir. 2007. Seam Carving for Content-Aware Image Resizing. In *ACM Trans. on Graphics (SIGGRAPH)*, Vol. 26(3). 10. <https://doi.org/10.1145/1275808.1276390>

[6] Hilda Azimi, Steven Chang, Jonathan Gold, and Koray Karabina. 2022. Improving Accuracy and Explainability of Online Handwriting Recognition. In *arXiv preprint arXiv:2209.09102*.

[7] Roman Bertolami and Horst Bunke. 2018. Hidden Markov Model-based Ensemble Methods for Offline Handwritten Text Line Recognition. In *Pattern Recognition*, Vol. 41(11). 3452–3460. <https://doi.org/10.1016/j.patcog.2008.04.003>

[8] Ali Furkan Biten, Andrés Mafla, Lluís Gómez, and Dimosthenis Karatzas. 2022. Is an Image Worth Five Sentences? A New Look into Semantics for Image-Text Matching. In *IEEE/CVF Winter Conf. on Applications of Computer Vision (WACV)*. Waikoloa, HI. <https://doi.org/10.1109/WACV51458.2022.00254>

[9] Théodore Bluche. 2016. Joint Line Segmentation and Transcription for End-to-End Handwritten Paragraph Recognition. In *Advances in Neural Information Processing Systems (NIPS)*. 838–846. <https://doi.org/10.5555/3157096.3157190>

[10] Théodore Bluche, Jérôme Louradour, and Ronaldo Messina. 2017. Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan. <https://doi.org/10.1109/ICDAR.2017.174>

[11] Théodore Bluche and Ronaldo Messina. 2017. Gated Convolutional Recurrent Neural Networks for Multilingual Handwriting Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan, 646–651. <https://doi.org/10.1109/ICDAR.2017.111>

[12] Hervé Bredin. 2017. TristouNet: Triplet Loss for Speaker Turn Embedding. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. 5430–5434. <https://doi.org/10.1109/ICASSP.2017.7953194>

[13] Andrew Brock, Jeff Donahue, and Karen Simonyan. 2019. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *Intl. Conf. on Learning Representations (ICLR)*.

[14] Matteo Bronkhorst. 2021. A Pen is All You Need. In *Twente Student Conf. on IT*. Enschede, The Netherlands.

[15] Yanling Bu, Lei Xie, Yafeng Yin, Chuyu Wang, Jingyi Ning, Jiannong Cao, and Sanglu Lu. 2021. Handwriting-Assistant: Reconstructing Continuous Strokes with Millimeter-level Accuracy via Attachable Inertial Sensors. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, Vol. 5(4), article 146. 1–25. <https://doi.org/10.1145/3494956>

[16] Victor Carbune, Pedro Gonnet, Thomas Deselaers, Henry A. Rowley, Alexander Daryin, Marcos Calvo, Li-Lun Wang, Daniel Keysers, Sandro Feuz, and Philippe Gervais. 2020. Fast Multi-language LSTM-based Online Handwriting Recognition. In *Intl. Journal on Document Analysis and Recognition (IJDAR)*, Vol. 23. 89–102. <https://doi.org/10.1007/s10032-020-00350-4>

[17] Dayvid Castro, Byron L. D. Bezerra, and Mêuser Valença. 2018. Boosting the Deep Multidimensional Long-Short-Term Memory Network for Handwritten Recognition Systems. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Niagara Falls, NY. <https://doi.org/10.1109/ICFHR-2018.2018.00031>

[18] Ushasi Chaudhuri, Biplab Banerjee, Avik Bhattacharya, and Mihai Datcu. 2020. CrossATNet - A Novel Cross-Attention based Framework for Sketch-based Image Retrieval. In *Image and Vision Computing*, Vol. 104. <https://doi.org/10.1016/j.imavis.2020.104003>

[19] Jiacheng Chen, Hexiang Hu, Hao Wu, Yuning Jiang, and CHanghu Wang. 2022. Learning the Best Pooling Strategy for Visual Semantic Embedding. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN. <https://doi.org/10.1109/CVPR46437.2021.01553>

[20] Junshan Kevin Chen, Wanze Xie, and Yutong (Kelly) He. 2021. Motion-based Handwriting Recognition. In *arXiv preprint arXiv:2101.06022*.

[21] Zhuo Chen, Yichao Wu, Fei Yin, and Cheng-Lin Liu. 2017. Simultaneous Script Identification and Handwriting Recognition via Multi-Task Learning of Recurrent Neural Networks. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan, 525–530. <https://doi.org/10.1109/ICDAR.2017.92>

[22] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a Similarity Metric Discriminatively, with Application to Face Verification. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. San Diego, CA. <https://doi.org/10.1109/CVPR.2005.202>

[23] Arindam Chowdhury and Lovekesh Vig. 2018. An Efficient End-to-End Neural Model for Handwritten Text Recognition. In *British Machine Vision Conference (BMVC)*.

[24] Shohreh Deldari, Hao Xue, Aaqib Saeed, Jiayuan He, Daniel V. Smith, and Flora D. Salim. 2022. Beyond Just Vision: A Review on Self-Supervised Representation Learning on Multimodal and Temporal Data. In *arXiv preprint arXiv:2206.02353*.

[45] Guozheng He, Zhouyi Wu, Yuting Wu, Peiying Lin, and Hiantao Huangfu. 2022. Online Handwriting Recognition Based on Microphone and IMU. In *Intl. Conf. on Electronics Technology (ICET)*. Chengdu, China. <https://doi.org/10.1109/ICET55676.2022.9824489>

[46] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV. <https://doi.org/10.1109/CVPR.2016.90>

[47] Qiang He, Zhiping Feng, Xue Wang, Yufen Wu, and Jin Wang. 2022. A Smart Pen Based on Triboelectric Effects for Handwriting Pattern Tracking and Biometric Identification. In *ACS Appl. Mater. Interfaces*, Vol. 14(43). <https://doi.org/10.1021/acsmami.2c13714>

[48] Alexander Hermans, Lucas Beyer, and Bastian Leibe. 2017. In Defence of the Triplet Loss for Person Re-Identification. In *arXiv preprint arXiv:1703.07737*.

[49] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2020. MHTN: Modal-Adversarial Hybrid Transfer Network for Cross-Modal Retrieval. In *Trans. on Cybernetics*, Vol. 50(3). 1047–1059. <https://doi.org/10.1109/TCYB.2018.2879846>

[50] Raashid Hussain, Ahsen Raza, Imran Siddiqi, Khurram Khurshid, and Chawki Djeddi. 2015. A Comprehensive Survey of Handwritten Document Benchmarks: Structure, Usage and Evaluation. In *EURASIP Journal on Image and Video Processing*, Vol. 46. <https://doi.org/10.1186/s13640-015-0102-5>

[51] R. Reeve Ingle, Yasuhisa Fujii, Thomas Deselaers, Jonathan Baccash, and Ashok C. Popat. 2019. A Scalable Handwritten Text Recognition System. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Sydney, Australia. <https://doi.org/10.1109/ICDAR.2019.00013>

[52] Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. 2022. ColloSSL: Collaborative Self-Supervised Learning for Human Activity Recognition. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*.

[53] Bo Ji and Tianyi Chen. 2020. Generative Adversarial Network for Handwritten Text. In *arXiv preprint arXiv:1907.11845*.

[54] Lei Kang, Pau Riba, Marcal Rusinol, Alicia Fornes, and Mauricio Villegas. 2022. Pay Attention to What You Read: Non-recurrent Handwritten Text-Line Recognition. In *Pattern Recognition*, Vol. 129. <https://doi.org/10.1016/j.patcog.2022.108766>

[55] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. 2020. Proxy Anchor Loss for Deep Metric Learning. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 3238–3247.

[56] Andreas Klaß, Sven M. Lorenz, Martin W. Lauer-Schmaltz, David Rügamer, Bernd Bischl, Christopher Mutschler, and Felix Ott. 2022. Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift. In *IJCAI-ECAI Intl. Workshop on Spatio-Temporal Reasoning and Learning (STR)*, Vol. 3190. Vienna, Austria.

[57] Fabian Kreß, Alexey Serdyuk, Tim Hotfilter, Julian Hoefer, Tanja Harbaum, Jürgen Becker, and Tim Hamann. 2022. Hardware-aware Workload Distribution for AI-based Online Handwriting Recognition in a Sensor Pen. In *Mediterranean Conference on Embedded Computing (MECO)*. Budva, Montenegro. <https://doi.org/10.1109/MECO55406.2022.9797131>

[58] Praveen Krishnan, Kartik Dutta, and C. V. Jawahar. 2018. Word Spotting and Recognition using Deep Embedding. In *IAPR Intl. Workshop on Document Analysis Systems (DAS)*. Vienna, Austria, 1–6. <https://doi.org/10.1109/DAS.2018.70>

[59] Hyodong Lee, Joonseok Lee, Joe Yue-Hei Ng, and Paul Natsev. 2020. Large Scale Video Representation Learning via Relational Graph Clustering. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA. <https://doi.org/10.1109/CVPR42600.2020.00684>

[60] Nan Li, Jinying Chen, Huaiyu Cao, Bing Zhang, and Prem Natarajan. 2014. Applications of Recurrent Neural Network Language Model in Offline Handwriting Recognition and Word Spotting. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Hersonissos, Greece. <https://doi.org/10.1109/ICFHR.2014.30>

[61] Wenbin Li, Xuesong Yang, Meihao Kong, Lei Wang, Jing Huo, Yang Gao, and Jiebo Luo. 2021. Triplet is All You Need with Random Mappings for Unsupervised Visual Representation Learning. In *arXiv preprint arXiv:2107.10419*.

[62] Dongyun Liang, Weiran Xu, and Ying Zhao. 2017. Combining Word-Level and Character-Level Representations for Relation Classification of Informal Text (RepL4NLP). In *Workshop on Representation Learning for NLP (RepL4NLP)*. Vancouver, Canada, 43–47. <https://doi.org/10.18653/v1/W17-2606>

[63] Jae Hyun Lim, Pedro O. O. Pinheiro, Negar Rostamzadeh, Chris Pal, and Sungjin Ahn. 2019. Neural Multisensory Scene Inference. In *Advances in Neural Information Processing Systems (NIPS)*, Vol. 32(807). 8996–9006. <https://doi.org/10.5555/3454287.3455094>

[64] Jae Hyun Lim and Jong Chul Ye. 2017. Geometric GAN. In *arXiv preprint arXiv:1705.02894*.

[65] Shikun Liu, Edward Johns, and Andrew J. Davison. 2019. End-to-End Multi-Task Learning with Attention. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, 1871–1880. <https://doi.org/10.1109/CVPR.2019.00197>

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss

244

Functions for Online Handwriting Recognition

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions

000:21

[66] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. SphereFace: Deep Hypersphere Embedding for Face Recognition. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI. <https://doi.org/10.1109/CVPR.2017.713>

[67] Marcus Liwicki and Horst Bunke. 2005. IAM-OnDB - an On-Line English Sentence Database Acquired from Handwritten Text on a Whiteboard. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Seoul, Korea, 956–961. <https://doi.org/10.1109/ICDAR.2005.132>

[68] Mingsheng Long, Yue Cao, Lianmin Wang, and Michael I. Jordan. 2015. Learning Transferable Features with Deep Adaptation Networks. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 37. 97–105. <https://doi.org/10.5555/3045118.3045130>

[69] Johannes Michael, Roger Labahn, Tobias Grüning, and Jochen Zöllner. 2019. Evaluating Sequence-to-Sequence Models for Handwritten Text Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Sydney, NSW. <https://doi.org/10.1109/ICDAR.2019.900208>

[70] Ahmadreza Momeni and Kedar Tatwawadi. 2018. Understanding LUPI (Learning Using Privileged Information). <https://web.stanford.edu/~kedart/files/lUPI.pdf>

[71] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. 2020. A Metric Learning Reality Check. In *Europ. Conf. on Computer Vision (ECCV)*. 681–699. https://doi.org/10.1007/978-3-030-58595-2_41

[72] Yasunori Ohishi, Marc Delcroix, Tsubasa Ochiai, Shoko Araki, Daiki Takeuchi, Daisuke Niizumi, Akisato Kimura, Noboru Harada, and Kunio Kashino. 2022. ConceptBeam: Concept Driven Target Speech Extraction. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 4252–4260. <https://doi.org/10.1145/3503161.3548397>

[73] Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. Lisboa, Portugal, 5934–5943. <https://doi.org/10.1145/3503161.3548167>

[74] Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. [Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition](#). In *IAPR Intl. Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS)*. Montreal, Canada.

[75] Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In *IEEE/CVF Winter Conf. for Applications on Computer Vision (WACV)*. Waikoloa, HI, 266–276. <https://doi.org/10.1109/WACV51458.2022.00131>

[76] Felix Ott, David Rügamer, Lucas Heublein, Tim Hamann, Jens Barth, Bernd Bischl, and Christopher Mutschler. 2022. Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In *International Journal on Document Analysis and Recognition (IJDAR)*, Vol. 25(12). 385–414. <https://doi.org/10.1007/s10032-022-00415-6>

[77] Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. 2020. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, Vol. 4(3), article 92. Cancún, Mexico. <https://doi.org/10.1145/3411842>

[78] Joan Pastor-Pellicer, Salvador Espa na Boquera, M. J. Castro-Bleda, and Francisco Zamora-Martínez. 2015. A Combined Convolutional Neural Network and Dynamic Programming Approach for Text Line Normalization. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Tunis, Tunisia, 341–345. <https://doi.org/10.1109/ICDAR.2015.7333780>

[79] Yuxin Peng, Xin Huang, and Yunzhen Zhao. 2017. An Overview of Cross-media Retrieval: Concepts, Methodologies, Benchmarks and Challenges. In *Trans. on Circuits and Systems for Video Technology*, Vol. 28(9). 2372–2385. <https://doi.org/10.1109/TCSVT.2017.2705068>

[80] Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. 2014. Dropout Improves Recurrent Neural Networks for Handwriting Recognition. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Hersonissos, Greece, 285–290. <https://doi.org/10.1109/ICFHR.2014.55>

[81] Rejean Plamondon and Sargur N. Srihari. 2000. On-line and Off-line Handwriting Recognition: A Comprehensive Survey. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 22(1). 63–84. <https://doi.org/10.1109/34.824821>

[82] Arik Poznanski and Lior Wolf. 2016. CNN-N-Gram for Handwriting Word Recognition. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, 2306–2314. <https://doi.org/10.1109/CVPR.2016.253>

[83] Joan Puigcerver. 2017. Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan, 67–72. <https://doi.org/10.1109/ICDAR.2017.20>

[84] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 139. 8748–8763.

[85] Viresh Ranjan, Nikhil Rasiwasia, and C. V. Jawahar. 2015. Multi-Label Cross-Modal Retrieval. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Santiago de Chile, Chile. <https://doi.org/10.1109/ICCV.2015.466>

[86] Hannes Rantzsch, Haojin Yang, and Christoph Meinel. 2016. Signature Embedding: Writer Independent Offline Signature Verification with Deep Metric Learning. In *Advances in Visual Computing (ISVC)*. 616–625. https://doi.org/10.1007/978-3-319-50832-0_60

[87] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Covello, Gabriel Doyle, Gert R.G. Lanckriet, Roger Levy, and Nuno Vasconcelos. 2010. A New Approach to Cross-Modal Multimedia Retrieval. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 251–260. <https://doi.org/10.1145/1873951.1873987>

[88] Nikolaos Sarafianos, Xiang Xu, and Ioannis A. Kakadiaris. 2019. Adversarial Representation Learning for Text-to-Image Matching. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Seoul, Korea, 5814–5824. <https://doi.org/10.1109/ICCV.2019.00059>

[89] Maximilian Schrapel, Max-Ludwig Stadler, and Michael Rohs. 2018. Pentelligence: Combining Pen Tip Motion and Writing Sounds for Handwritten Digit Recognition. In *Proc. of the CHI Conf. on Human Factors in Computing Systems*. 1–11. <https://doi.org/10.1145/3173574.3173705>

[90] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA. <https://doi.org/10.1109/CVPR.2015.7298682>

[91] David Semedo and João Magalhães. 2020. Adaptive Temporal Triplet-loss for Cross-modal Embedding Learning. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 1152–1161. <https://doi.org/10.1145/3394171.3413540>

[92] Annapurna Sharma, Rahul Ambati, and Dinesh Babu Jayagopi. 2020. Towards Faster Offline Handwriting Recognition using Temporal Convolutional Networks. In *NCVPRIPG 2019 Communications in Computer and Information Science (CCIS)*, Springer, Singapore, Vol. 1249. 344–354. https://doi.org/10.1007/978-981-15-8697-2_32

[93] Annapurna Sharma and Dinesh Babu Jayagopi. 2021. Towards Efficient Unconstrained Handwriting Recognition using Dilated Temporal Convolutional Network. In *Expert Systems with Applications*, Vol. 164. <https://doi.org/10.1016/j.eswa.2020.114004>

[94] Baoguang Shi, Xiang Bai, and Cong Yao. 2017. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 39(11). 2298–2304. <https://doi.org/10.1109/TPAMI.2016.2646371>

[95] Shashank Kumar Singh and Amrita Chaturvedi. 2023. Leveraging Deep Feature Learning for Wearable Sensors Based Handwritten Character Recognition. In *Biomedical Signal Processing and Control*, Vol. 80(1). <https://doi.org/10.1016/j.bspc.2022.104198>

[96] Sumeet S. Singh and Sergey Karayev. 2021. Full Page Handwriting Recognition via Image to Sequence Extraction. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Lausanne, Switzerland, 55–69. https://doi.org/10.1007/978-3-030-86334-0_4

[97] Sebastian Sudholt and Gernot A. Fink. 2018. Attribute CNNs for Word Spotting in Handwritten Documents. In *Intl. Journal on Document Analysis and Recognition (IJDAR)*, Vol. 21. 199–218. <https://doi.org/10.1007/s10032-018-0295-0>

[98] Jorge Sueiras, Victoria Ruiz, Angel Sanchez, and Jose F. Velez. 2018. Offline Continuous Handwriting Recognition Using Sequence-to-Sequence Neural Networks. In *Neurocomputing*, Vol. 289(C). 119–128. <https://doi.org/10.1016/j.neucom.2018.02.008>

[99] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going Deeper with Convolutions. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, 1–9. <https://doi.org/10.1109/CVPR.2015.7298594>

[100] Bing Tian, Yong Zhang, Jin Wang, and Chunxiao Xing. 2019. Hierarchical Inter-Attention Network for Document Classification with Multi-Task Learning. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*. 3569–3575. <https://doi.org/10.24963/ijcai.2019/495>

[101] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. In *Journal of Machine Learning Research (JMLR)*, Vol. 9(86). 2579–2605.

[102] Vladimir Vapnik and Rauf Izmailov. 2015. Learning Using Privileged Information: Similarity Control and Knowledge Transfer. In *Journal of Machine Learning Research (JMLR)*. 2023–2049. <https://doi.org/10.5555/2789272.2886814>

[103] Shashanka Venkataraman, Ewa Kijak, Laurent Amsaleg, and Yannis Avrithis. 2022. AlignMix: Improving Representations by Interpolating Aligned Features. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 19174–19183.

[104] Alessandro Vinciarelli and Michael Peter Perrone. 2003. Combining Online and Offline Handwriting Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Edinburgh, UK, 844–848. <https://doi.org/10.1109/ICDAR.2003.1227781>

[105] Paul Voigtlaender, Patrick Doetsch, and Hermann Ney. 2016. Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*.

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition

246

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions

000:23

Shenzhen, China, 228–233. <https://doi.org/10.1109/ICFHR.2016.0052>

- [106] Qian Wan and Qin Zou. 2021. Learning Metric Features for Writer-Independent Signature Verification using Dual Triplet Loss. In *Intl. Conf. on Pattern Recognition (ICPR)*. Milan, Italy, 3853–3859. <https://doi.org/10.1109/ICPR48806.2021.9413091>
- [107] Junsheng Wang, Tiantian Gong, Zhixiong Zeng, Changchang Sun, and Yan Yan. 2022. C³CMR: Cross-Modality Cross-Instance Contrastive Learning for Cross-Media Retrieval. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 4300–4308. <https://doi.org/10.1145/3503161.3548263>
- [108] Jeen-Shing Wang, Yu-Liang Hsu, and Cheng-Ling Chu. 2013. Online Handwriting Recognition Using an Accelerometer-Based Pen Device. In *Intl. Conf. on Computer Science and Engineering (CSE)*. <https://doi.org/10.2991/cse.2013.52>
- [109] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Canjie Luo, Xiaoxue Chen, Yaqiang Wu, Qianying Wang, and Mingxiang Cai. 2020. Decoupled Attention Network for Text Recognition. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 34(7). 12216–12224. <https://doi.org/10.1609/aaai.v34i07.6903>
- [110] Zhiguang Wang and Tim Oates. 2015. Imaging Time-Series to Improve Classification and Imputation. In *Intl. Joint. Conf. on Artificial Intelligence (IJCAI)*. Buenos Aires, Argentina, 3939–3945. <https://doi.org/10.5555/2832747.2832798>
- [111] Lukas Wegmeth, Alexander Hoelzemann, and Kristof Van Laerhoven. 2021. Detecting Handwritten Mathematical Terms with Sensor Based Data. In [arXiv preprint arXiv:2109.05594](https://arxiv.org/abs/2109.05594).
- [112] Yunchao Wei, Yao Zhao, Canyi Lu, Shikui Wei, Luoqi Liu, Zhenfeng Zhu, and Shuicheng Yan. 2016. Cross-Modal Retrieval with CNN Visual Features: A New Baseline. In *Trans. on Cybernetics*, Vol. 47(2). 449–460. <https://doi.org/10.1109/TCYB.2016.2519449>
- [113] Kilian Q. Weinberger, John Blitzer, and Lawrence K. Saul. 2005. Distance Metric Learning for Large Margin Nearest Neighbor Classification. In *Advances in Neural Information Processing Systems (NIPS)*. 1473–1480. <https://doi.org/10.5555/2976248.2976433>
- [114] Curtis Wigington, Seth Stewart, Brian Davis, Bill Barrett, Brian Price, and Scott Cohen. 2017. Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan. <https://doi.org/10.1109/ICDAR.2017.110>
- [115] Yijung Xiao and Kyunghyun Cho. 2016. Efficient Character-level Document Classification by Combining Convolution and Recurrent Layers. In [arXiv preprint arXiv:1602.00367](https://arxiv.org/abs/1602.00367).
- [116] Tomoki Yoshida, Ichiro Takeuchi, and Masayuki Karasuyama. 2019. Safe Triplet Screening for Distance Metric Learning. In *Neural Computation*, Vol. 31(12). 2432–2491. https://doi.org/10.1162/neco_a_01240
- [117] Mohamed Yousef and Tom E. Bishop. 2020. OrigamiNet: Weakly-Supervised, Segmentation-Free, One-Step, Full Page Text Recognition by Learning to Unfold. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, 14710–14719. <https://doi.org/10.1109/CVPR42600.2020.01472>
- [118] Mohamed Yousef, Khaled F. Hussain, and Usama S. Mohammed. 2020. Accurate, Data-Efficient, Unconstrained Text Recognition with Convolutional Neural Networks. In *Pattern Recognition*, Vol. 108.
- [119] Donghuo Zeng, Yi Yu, and Keizo Oyama. 2020. Deep Triplet Neural Networks with Cluster-CCA for Audio-Visual Cross-Modal Retrieval. In *ACM Trans. on Multimedia Computing, Communications, and Applications*, Vol. 16(3). 1–23. <https://doi.org/10.1145/3387164>
- [120] Xiangsheng Zeng, Donglai Xiang, Liangrui Peng, Changsong Liu, and Xiaoqing Ding. 2017. Local Discriminant Training and Global Optimization for Convolutional Neural Network Based Handwritten Chinese Character Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan. <https://doi.org/10.1109/ICDAR.2017.70>
- [121] Dawei Zhang and Zhonglong Zheng. 2020. Joint Representation Learning with Deep Quadruplet Network for Real-Time Visual Tracking. In *IEEE Intl. Joint Conf. on Neural Networks (IJCNN)*. Glasgow, UK. <https://doi.org/10.1109/IJCNN48605.2020.9207185>
- [122] Hongyu Zhang, Lichang Chen, Yunhao Zhang, Renjie Hu, Chunjuan He, Yaqing Tan, and Jiajin Zhang. 2022. A Wearable Real-Time Character Recognition System Based on Edge Computing-Enabled Deep Learning for Air-Writing. In *Journal of Sensors*, Vol. 2022. <https://doi.org/10.1155/2022/8507706>
- [123] Ji Zhang, Yannis Kalantidis, Marcus Rohrbach, Ahmed Elgammal, and Mohamed Elhoseiny. 2019. Large-Scale Visual Relationship Understanding. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 33(1). 9185–9194. <https://doi.org/10.1609/aaai.v33i01.33019185>
- [124] Yi Zhen, Piyush Rai, Hongyuan Zha, and Lawrence Carin. 2015. Cross-Modal Similarity Learning via Pairs, Preferences, and Active Supervision. In *Association for the Advancement of Artificial Intelligence (AAAI)*. 3203–3209. <https://doi.org/10.5555/2888116.2888162>

A APPENDICES

We provide more information about the broader impact, limitations, ethical concerns, and a comparison to writing on touch screen surfaces in Section A.1. While Section A.2 gives an overview of methods for offline handwriting recognition, Section A.3 summarizes cross-modal retrieval methods, the corresponding modalities, pairwise learning, and deep metric learning. We present the multi-task learning technique in Section A.4, and show more details on learning with the triplet loss on synthetically generated signal and image data in Section A.5. We propose more details of our architectures in Section A.6. Section A.7 presents results of representation learning for online HWR.

A.1 Statements

Broader Impact Statement. While research for offline handwriting recognition (HWR) is well-established, research for online HWR from sensor-enhanced pens only emerged in 2019. Hence, the methodological research for online HWR currently does not meet the requirements for real-world applications. Handwriting is still important in different fields, in particular graphomotoricity as a fine motor skill. The visual feedback provided by the pen helps young students to learn a new language. A well-known bottleneck for many machine learning algorithms is their requirement for large amounts of datasets, while data recording of handwriting data is time-consuming. This paper extends the online HWR dataset with generated images from offline handwriting and closes the gap between offline and online HWR by using offline HWR as an auxiliary task by learning with privileged information. One downside of training the offline architecture (consisting of gated text recognizer blocks) is its long training time. However, as this model is not required at inference time, processing the time-series is still fast. The cross-modal representation between both modalities (image and time-series) is achieved by using the triplet loss and a sample selection depending on the Edit distance. This approach is important in many applications of sequence-based classification, i.e., the triplet loss evolved recently for language processing applications such as visual semantic clustering, while pairwise learning is typically applied in fields such as image recognition. Ethical statement about collection consent and personal information: For data recording, the consent of all participants was collected. The datasets only contain the raw data from the sensor-enhanced pen and – for statistics – the age, gender, and handedness of the participants. The datasets are fully pseudonymized by assigning an ID to every participant. The datasets do not contain any personal identifying information. The approach proposed in this paper – in particular, when used for the application of online handwriting recognition from sensor-enhanced pens – does not (1) facilitate injury to living beings, (2) raise safety or security concerns (due to the anonymity of the data), (3) raise human rights concerns, (4) have a detrimental effect on people’s livelihood, (5) develop harmful forms of surveillance (as the data is pseudonymized), (6) damage the environment, and (7) deceive people in ways that cause harm.

Limitations. The limitation of the method is the requirement of multiple image-based datasets in the same language. As the OnHW-words and OnHW-wordsRandom datasets are written in German and contain word labels with mutated vowels, a similar image-based German dataset is required, which does not currently exist. The available dataset most similar to the OnHW dataset is the IAM-OffDB dataset, which does not contain mutated vowels. Hence, the OCR method cannot be pre-trained on words with mutated vowels. In conclusion, the method is not limited by ScrabbleGAN, but by the image-based dataset required for pre-training. The gated text recognizer could also be directly pre-trained on the IAM-OffDB dataset, but we assume less generalized results than for our generated dataset.

Statement on Ethical Concerns. Machine learning models face various challenges when classifying text with this sensor-enhanced pen. These challenges can appear if there is a domain shift between training and test datasets, e.g., specific writers have a unique writing style and accelerations, or they hold the pen differently. Also, some writers might have a unique writing environment (different writing surfaces such as a unique table or paper which leads to different magnetic fields). Another difficulty can appear through an under-represented group such as left-handed writers or a disabled person for which the model is not trained on. A well-generalized model trained on all possible pen movements is very challenging and requires a lot of training data. One solution is to record data for a unique writer and adapt the model, or augment the data for a better representation, e.g., as proposed with our method on left-handed writers. Hence, unique writers are not excluded and the task for classifying writing from under-represented groups is addressed in our paper, but domain shifts still remain a challenging problem.

Comparison to Writing on Touch Screen Surfaces. Methods for writing on surfaces such as the iPad OS system and others require a tablet with a touch screen surface and stylus pens with integrated magnetometers or pressure sensitivity. These methods can easily reconstruct the trajectory of the pen tip through the magnetometer on the surface, and hence, can classify the written text. This is more challenging when using sensor-enhanced pens, as the classification task is performed directly on the sensor data. One drawback of methods used in the iPad OS is the requirement for writing on specific surfaces, which in turn can influence the writing style. Also, certain applications require writing on normal paper, or the availability of a touch screen surface is not always given, e.g., when writing a short list, but notes need to be digitized afterwards.

A.2 Offline Handwriting Recognition

In the following, we give a detailed overview of offline HWR methods to select a suitable lexicon and language model-free method. To our knowledge, there is no recent paper summarizing published work for offline HWR. For an overview of offline and online HWR datasets, see [54, 85]. Table 7 presents related work. Methods for offline HWR range from hidden Markov models (HMMs) to deep learning techniques that became predominant in 2014, such as convolutional neural networks (CNNs), temporal convolutional networks (TCNs), and recurrent neural networks (RNNs). RNN techniques are well explored, including long short-term memories (LSTMs), bidirectional LSTMs (BiLSTMs), and multidimensional RNNs (MDRNN, MDLSTM). Recent methods are generative adversarial networks (GANs) and Transformers. In Table 7, we refer to the use of a language model as LM with k and identify the data level on which the method works – i.e., paragraph or full-text level (P), line level (L), and word level (W). We present evaluation results for the IAM-OffDB [73] and RIMES [43] datasets. We show the character error rate (CER) – the percentage of characters that were incorrectly predicted (the lower, the better) – and the word error rate (WER) – a common performance metric on word level instead of the phoneme level (the lower, the better).

HMMs. In the past, various methods based on HMMs have been proposed [6, 32, 67, 81]. Recently, [34] proposed HMM+ANN, an HMM modeled with Markov chains in combination with a multilayer perceptron (MLP) to estimate the emission probabilities. [62] presented Tandem GHMM that uses moment-based image normalization, writer adaptation, and discriminative feature extraction with a 3-gram open-vocabulary of size 50k with an LSTM for recognition. [31] proposed an LSTM unit that controls the shape of the squashing function in gating units decoded in a hybrid HMM. This approach yields the best results based on HMMs.

RNNs: MDLSTMs. The 2DLSTM approach by [42] combines multidimensional LSTMs (MDLSTMs) with the CTC loss. The MDLSTM-RNN approach [10] works at paragraph level by replacing

Table 7. Evaluation results (WER and CER in %) of different methods on the IAM-OffDB [73] and RIMES [43] datasets. The table is sorted by year.

Method		Information	LM size k	Level	IAM-OffDB		RIMES	
				P L W	WER	CER	WER	CER
HMM	HMM+ANN [34]	Markov chain with MLP	w/ (5)		15.50	6.90	-	-
	Tandem GHMM [62]	GHMM and LSTM, writer adaptation	w/ (50)	×	13.30	5.10	13.70	4.60
	LSTM-HMM [31]	Combination of LSTM with HMM	w/ (50)	×	12.20	4.70	12.90	4.30
	2DLSTM [42]	Combined MDLSTM with CTC	w/o		27.50	8.30	17.70	4.00
	MDLSTM-RNN [10]	150 dpi	w/o	×	29.50	10.10	13.60	3.20
		150 dpi	w/ (50)	×	16.60	6.50	-	-
		300 dpi	w/o	×	24.60	7.90	12.60	2.90
		300 dpi	w/ (50)	×	16.40	5.50	-	-
		GPU-based, diagonal MDLSTM			9.30	3.50	9.60	2.80
		Multi-task approach	w/o		34.55	11.15	30.54	8.29
Multi-dim. LSTM	SepMDLSTM [22]	MDLSTM, attention	w/o	×	-	16.20	-	-
	[11]	Line segmentation 150 dpi	w/o	×	-	11.10	-	-
		Line segmentation 150 dpi	w/o	×	-	7.50	-	-
			w/ (20)		10.50	3.60	-	-
					18.20	25.90	-	-
	MDLSTM [16]	Sliding win. Gaussian HMM, RNN	w/o	×	35.10	10.80	28.50	6.80
	BiLSTM [41]	LSTMs with dropout	w/o		12.70	4.80	12.10	4.40
	HMM+RNN [75]	Maximum mutual information	w/ (50)		10.90	4.40	11.20	3.50
	Dropout [83]				13.60	5.10	12.30	3.30
	[106]				10.50	3.20	7.90	1.90
RNN	[9]				18.40	5.80	9.60	2.30
	GCRNN [12]	CNN+BiLSTM	w/ (50)		12.20	4.40	9.00	2.50
	CNN-1DLSTM-CTC [87]	CNN+BiLSTM+CTC (128 \times width)	w/o	×	16.19	6.34	-	-
		NN+BiLSTM+CTC	w/ (50)		32.89	9.78	-	-
	End2End [63]	Without line level	w/		23.20	6.40	9.30	2.10
		Line level	w/o	×	12.61	4.88	7.04	2.32
	SFR [115]	Text detection and segmentation	w/		4.80	2.52	1.86	0.65
	CNN-RNN [33]	Unconstrained	w/o	×	4.07	2.17		
		Full-Lexicon	w/o		17.82	5.70	9.60	2.30
		Text-Lexicon	w/		25.50	17.40	19.10	12.00
CNN	[24]	Unconstrained	w/o	×	22.90	13.10	15.80	9.70
		Seq2seq, w/o LN	w/o		21.10	11.40	13.50	7.30
		w/ LN	w/o		16.70	8.10	9.60	3.50
		w/ LN + Focal Loss	w/o		15.90	4.80	-	-
	[100]	w/ LN + Focal Loss + Beam Search	w/	×	-	8.50	-	-
	[26]	LSTM encoder-decoder, attention			30.70	12.80	-	-
	[55]	ResNet+LSTM, segmentation			35.20	14.10	-	-
		BiLSTM			-	5.24	-	-
		GRCL			-	15.60	-	-
GAN	[77]	Seq2seq CNN+BiLSTM (64 \times width)	w/		8.87	5.94	6.31	3.17
	FPN [14]	Feature Pyramid Network, 150 dpi	w/		6.45	3.44	3.90	1.90
	AFDM [7]	AFD module	w/o					
	[86]	CNN + connected branches, CCA	w/o		4.90	-	-	-
		CNN+CTC (32 \times width)	w/o	×	51.37	-	-	-
		VGG (500 \times 500)	w/o	×	34.55	-	-	-
		VGG (500 \times 500), w/o LN	w/o	×	10.03	-	-	-
		ResNet26 (500 \times 500), w/o LN	w/o	×	7.24	-	-	-
		ResNet26 (500 \times 500), w/o LN	w/o	×	8.93	-	-	-
		ResNet26 (500 \times 500), w/ LN	w/o	×	6.37	-	-	-
Trans-former		ResNet26 (500 \times 500), w/o LN	w/o	×	76.90	-	-	-
		ResNet26 (500 \times 500), w/ LN	w/o	×	6.13	-	-	-
		GTR-8 (500 \times 500), w/o LN	w/o	×	72.40	-	-	-
		GTR-8 (500 \times 500), w/ LN	w/o	×	5.64	-	-	-
		GTR-8 (750 \times 750), w/ LN	w/o	×	5.50	-	-	-
		GTR-12 (750 \times 750), w/ LN	w/o	×	4.70	-	-	-
	DAN [111]	Decoupled attention module	w/o	×	19.60	6.40	8.90	2.70
	ScrabbleGAN [37]	Original data	w/o		25.10	-	12.29	-
		Augmentation	w/o		24.73	-	12.24	-
		Augmentation + 100k synth.	w/o		23.98	-	11.68	-
Other	[59]	Augmentation + 100k synth. + Refine	w/o		23.61	-	11.32	-
	FPHR [97]	Self-attention for text/images	w/o	×	15.45	4.67	-	-
		CNN encoder, Transformer decoder	w/o	×	-	6.70	-	-
Other	FST [76]	With augmentation	w/o	×	-	6.30	-	-
		Finite state transducer (lexicon)	n-gram		19.10	-	13.30	-

Abbreviations. Size k of the language model (LM), i.e., with (w/) or without (w/o) a LM. P: paragraph or full text level, L: line level, LN: layer normalization, CER: character error rate, WER: word error rate, MLP: multi-layer perceptron, HMM: hidden markov model, GTR: gated text recognizer, seq2seq: sequence-to-sequence, GAN: generative adversarial network, CTC: connectionist temporal classification, RNN: recurrent neural network, LSTM: long short-term memory, CNN: convolutional neural network

the collapse layer with a recurrent version. A neural network performs implicit line segmentation by computing attention weights on the image representation. [105] proposed an efficient GPU-based implementation of MDLSTMs by processing the input in a diagonal-wise fashion. SepMDLSTM [22] is a multi-task learning method for script identification and HWR based on two classification modules by minimizing the CTC and negative log-likelihood losses. While the MDLSTM by [11] contains covert and overt attention without prior segmentation, the [16] integrated MDLSTMs within a hybrid HMM. However, these architectures come with expensive computational cost. Furthermore, they extract features visually similar to those of convolutional layers [87]. End2End [63] jointly learns text and image embeddings based on LSTMs.

RNNs: LSTMs and BiLSTMs. RNNs for HWR marked an important milestone in achieving impressive recognition accuracies. Sequential architectures are perfect to fit text lines, due to the probability distributions over sequences of characters, and due to the inherent temporal aspect of text [59]. [41] introduced the BiLSTM layer in combination with the CTC loss. [83] showed that the performance of LSTMs can be greatly improved using dropout. [106] investigated sequence-discriminative training of LSTMs using the maximum mutual information (MMI) criterion. While [9] utilized an RNN with an HMM and a language model, [75] combined an RNN with a sliding window Gaussian HMM. GCRNN [12] combines a convolutional encoder (aiming for generic and multilingual features) and a BiLSTM decoder predicting character sequences. Additionally, [87] proposed a CNN+BiLSTM architecture (CNN-1DLSTM-CTC) that uses the CTC loss. The start, follow, read (SFR) [115] model jointly learns text detection and segmentation. [33] used synthetic data for pre-training and image normalization for slant correction. The methods by [24, 55, 77, 100] also make use of BiLSTMs. While [14] uses a feature pyramid network (FPN), the adversarial feature deformation module (AFDM) [7] learns ways to elastically warp extracted features in a scalable manner. Further methods that combine CNNs with RNNs are [69, 99, 117], while BiLSTMs are utilized in [15, 102].

TCNs. TCNs use dilated causal convolutions and have been applied to air-writing recognition by [5]. As RNNs are slow to train, [94] presented a faster system that is based on text line images and TCNs with the CTC loss. This method achieves 9.6% CER on the IAM-OffDB dataset. [95] combined 2D convolutions with 1D dilated non-causal convolutions that offer high parallelism with a smaller number of parameters. They analyzed re-scaling factors and data augmentation and achieved comparable results for the IAM-OffDB and RIMES datasets.

CNNs. [86] utilized a CNN with multiple fully connected branches to estimate its n-gram frequency profile (set of n-grams contained in the word). With canonical correlation analysis (CCA), the estimated profile can be matched to the true profiles of all words in a large dictionary. As most attention methods suffer from an alignment problem, [111] proposed a decoupled attention network (DAN) that has a convolutional alignment module that decouples the alignment operation from using historical decoding results based on visual features. The gated text recognizer [121] aims to automate the feature extraction from raw input signals with a minimum required domain knowledge. The fully convolutional network without recurrent connections is trained with the CTC loss. Thus, the gated text recognizer module can handle arbitrary input sizes and can recognize strings with arbitrary lengths. This module has been used for OrigamiNet [120] which is a segmentation-free multi-line or full-page recognition system. OrigamiNet yields state-of-the-art results on the IAM-OffDB dataset, and shows improved performance of gated text recognizer over VGG and ResNet26. Hence, we use the gated text recognizer module as our visual feature encoder for offline HWR (see Section A.6).

GANs. Handwriting text generation is a relatively new field. The first approach by [40] was a method to synthesize online data based on RNNs. The technique HWGAN by [58] extends this

Table 8. Overview of cross-modal and pairwise learning techniques using the modalities video (V), images (I), audio (A), text (T), sensors (S), or haptic (H). Data from sensors are represented by time-series from inertial, biological, or environmental sensors. We indicate cross-modal learning from the same modality with \times^n with n modalities. If n is unspecified, the method can potentially work with an arbitrary number of modalities.

Method (sorted by year)	Modality						Pairwise Learning	Deep Metric Learning Loss/Objective	Application
	V	I	A	T	S	H			
[23]	\times^2						Pairwise	L_1 similarity	Face verification
[90]	\times		\times				Pairwise	Canonical correlation analysis	Multimedia documents: emb. mapping to common space
DeViSE [38]	\times	\times					Hinge rank	Cosine similarity	Visual semantic embedding
OxfordNet [61]	\times	\times					Contrastive	Cosine similarity	Visual semantic embedding
[119]	\times	\times					Denotation graph	Pointwise MI	Visual semantic embedding
DAN [74]	\times^2						Pairwise	Kernelized MMD	Domain adaptation
ml-CCA [89]	\times	\times					Not pairwise	CCA extended	Multi-label annotations
FaceNet [92]	\times						Triplet	Euclidean	Face recognition, clustering
deep-SM [114]	\times^2		\times				Pairwise	CCA, T-V CCA semantic matching	Universal representation for various recognition tasks
[39]	\times	\times					Triplet	non-Mercer match kernel	Visual semantic embedding
TristouNet [13]		\times					Triplet	Euclidean	Speech classification
Triplet+FANNG [50]	\times						Smart triplet	Nearest neighbour graph	General
[123]	\times						Triplet	CE, conditional random field	Handwritten Chinese characters recognition
[53]	\times	\times					Pairwise	Cosine similarity	Visual semantic embedding
GXN [44]	\times	\times					Triplet	Similarity: order-violation penalty	Visual semantic embedding
TDH [28]	\times^2		\times				Triplet	Hamming space	Visual semantic embedding
VSE++ [35]	\times	\times					Triplet	Similarity: inner product	Visual semantic embedding
SCAN t-i [65]	\times	\times					Triplet	Similarity LSE	Visual semantic embedding
Discriminative [30]	\times						Triplet	Class centroids	Image classification
VSRN [66]	\times	\times	\times				Triplet	Similarity: inner product	Visual semantic embedding
PIE-Nets [98]	\times	\times	\times				Pairwise	Diversity, MIL, MMD	Visual semantic embedding
LIWE [113]	\times	\times					Contrastive	Sum/Max of Hinges	Visual semantic embedding
[124]	\times						STriplet+triplet	Cosine similarity	Relationship understanding
TIMAM [91]	\times	\times	\times				Pairwise	Norm-softmax CE	Visual question answering
GMN [70]	\times^n	\times^n			\times^n		Pairwise	Cross-modal generation	Multisensory 3D scenes
CTM [46]	\times		\times				Triplet	CTC, CE, L_2 correlation	Sentence translation
UniVSE [116]	\times	\times					Contrastive	Alignment losses	Visual semantic embedding
ActiveSet+RRPB [118]	\times						Smart triplet	Semidefinite constraint	General
PAN [125]	\times	\times					Pairwise	Cosine similarity	Visual semantic embedding
CM-GANs [82]	\times	\times					Adversarial	Inter/intra class	Visual semantic embedding
CPC [103]	\times	\times		\times			Contrastive	CE, MI	One modality classification
CrossATNet [17]	\times^2		\times				Triplet	MSE	Zero-shot learning, sketches
MHTN [52]	\times	\times	\times				Pairwise, contr.	MMD, Euclidean	CMR
GCML [64]	\times^2		\times				Triplet	Hierarchical relational graph clustering	Retrieval, search, video-to-video similarity
CSVE [108]	\times		\times				Bidirect. triplet	Correlation graph	Visual semantic embedding
TXS-Adapt [93]	\times		\times				Triplet	Recency-based correlation	Social media domain
Proxy-Anchor [60]	\times						Pair+proxy	Cosine similarity	Image classification
TNN-C-CCA [122]	\times	\times					Triplet	CCA	Multimedia
AdapOffQuin [21]	\times		\times				Quintuplet	Cosine similarity	Visual semantic embedding
ROMA [68]	\times^2						Soft triplet	CE, random perturbation	Unsupervised representation learning
CLIP [88]	\times	\times					Contrastive	CE, Cosine similarity	OCR, action/object recognition
SGRAF [29]	\times	\times					Pairwise	Vector similarity	Visual semantic embedding
PCME [25]	\times		\times				Triplet	Euclidean	Visual semantic embedding
MCN [18]	\times	\times	\times				Contrastive	Similarity, reconstruction	Multimodal clustering
VATT [1]	\times	\times	\times				Contrastive	CC, NCE, MIL-NCE	Transformer for CMR
[107]	\times						Dual triplet	Euclidean	Signature verification
[110]	\times		\times^2				Contrastive	Cosine similarity	Audio classification
[48]	\times^2						Triplet	Softmax, MSE	Person re-identification
AlignMixup [104]	\times^2						Pairwise	Sinkhorn transport	Data augmentation for interpolation

Table 9. Table 8 continued.

Method (sorted by year)	V	I	Modality A	T	S	H	Pairwise Learning	Deep Metric Learning Loss/Objective	Application
SAM [8]	×		×				Triplet	Cosine similarity	Visual semantic embedding
VSE _∞ [20]	×		×				Triplet	Similarity	Visual semantic embedding
AudioCLIP [47]	×	×	×				Contrastive	Cosine similarity, symmetric CE	Environmental sound classification
data2vec [4]	×	×	×				Predicts latent representations		Self-supervision with masks
ColloSSL [57]						ⁿ	Contrastive	CE, Cosine similarity	Human-activity recognition
COCOA [27]						ⁿ	Contrastive	Cosine similarity	General time-series
ELO [84] [71]	×	×	×	×			Contrastive	L_2 , evolutionary	Cross-modal, multi-task Crowd counting
MM-ALT [45]	×	×		×			Pairwise	-	Lyric transcription
FLAVA [96]							Pairwise	CTC, residual attention	Visual semantic embedding
ConceptBeam [78] [36]						ⁿ	Contrastive	Cosine similarity, temperature scaling	Target speech extraction Text-video retrieval, kitchen
C ³ CMR [109] [19]	×						Triplet	Cosine similarity	Visual semantic embedding VSE with graph embedding
[79]	×						Contrastive	-	Online HWR
CMR-IS (Ours, 2022)	×					²	Classwise	CE, HoMM/CC/PC	Online HWR
							Contr., triplet	CTC, MSE/CC/PC/KL	Online HWR

Abbreviations. CE: cross-entropy, CTC: connectionist temporal classification, MSE: mean squared error, CC: cross-correlation, PC: Pearson correlation, MMD: maximum mean discrepancy, HoMM: higher-order moment matching, CCA: canonical correlation analysis, MIL: multiple-instance learning, MI: mutual information, NCE: noise contrastive estimation, VSE: visual semantic embedding

method by adding a discriminator \mathcal{D} . DeepWriting [2] is a GAN that is capable of disentangling style from content and thus making digital ink editable. [49] proposed a method to generate handwriting based on a specific author with learned parameters for spacing, pressure, and line thickness. [3] used a BiLSTM to obtain an embedding of the word to be rendered and added an auxiliary network as a recognizer \mathcal{R} . The model is trained with a combination of an adversarial loss and the CTC loss. ScrabbleGAN by [37] is a semi-supervised approach that can arbitrarily generate many images of words with arbitrary length from a generator \mathcal{G} to augment handwriting data and uses a discriminator \mathcal{D} and recognizer \mathcal{R} . The paper proposes results for original data with random affine augmentation using synthetic images and refinement.

Transformers. RNNs prevent parallelization, due to their sequential pipelines. [59] introduced a non-recurrent model by the use of Transformer models with multi-head self-attention layers at the textual and visual stages. Their method works for any pre-defined vocabulary. For the feature encoder, they used modified ResNet50 models. The full page HTR (FPHR) method by [97] uses a CNN as an encoder and a Transformer as a decoder with positional encoding.

A.3 Overview of Cross-Modal Retrieval Methods

We provide a summary of methods for cross-modal learning in Table 8 and Table 9. Typical modalities are video, image, audio, text, sensors (such as inertial sensors used for our method), and haptic modalities. We classify each method with the technique used for pairwise learning that utilizes an objective for deep metric learning. The overview contains a wide range of applications, while visual semantic embedding is a common field for cross-modal retrieval.

A.4 Multi-Task Learning

We simultaneously train the \mathcal{L}_{CTC} loss for sequence classification combined with one or two shared losses $\mathcal{L}_{shared,1}$ and $\mathcal{L}_{shared,2}$ for cross-modal representation learning. As both losses are in different ranges, the naive weighting

$$\mathcal{L}_{total} = \sum_{i=1}^{|T|} \omega_i \mathcal{L}_i, \quad (3)$$

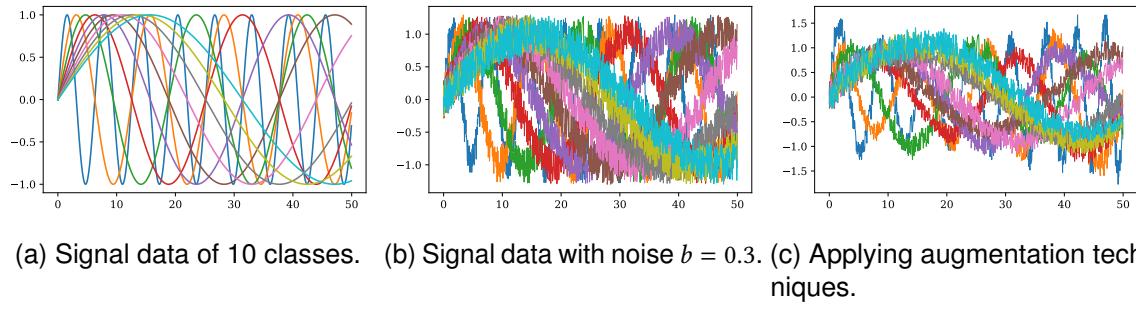


Fig. 10. Plot of the 1D signal data for 10 classes.

with pre-specified constant weights $\omega_i = 1, \forall i \in \{1, \dots, |T|\}$ can harm the training process. Hence, we apply dynamic weight average (DWA) [72] as a multi-task learning approach that performs dynamic task weighting over time (i.e., after each batch).

A.5 Training Synthetic Data with the Triplet Loss

Signal and Image Generation. We combine the networks for both signal and image classification to improve the classification accuracy over each single-modal network. The aim is to show that the triplet loss can be used for such a cross-modal setting in the field of cross-modal representation learning. Hence, we generate synthetic data in which the image data contains information of the signal data. We generate signal data \mathbf{x} with $x_{i,k} = \sin(0.05 \cdot \frac{t_i}{k})$ for all $t_i \in \{1, \dots, 1,000\}$ where t_i is the timestep of the signal. The frequency of the signal is dependent on the class label k . We generate signal data for 10 classes (see Figure 10a). We add noise from a continuous uniform distribution $U(a, b)$ for $a = 0$ and $b = 0.3$ (see Figure 10b) and add time and magnitude warping (see Figure 10c). We generate a signal-image pair such that the image is based on the signal data. We make use of the Gramian angular field that transforms time-series into images. The time-series is defined as $\mathbf{x} = (x_1, \dots, x_n)$ for $n = 1,000$. The Gramian angular field creates a matrix of temporal correlations for each (x_i, x_j) by rescaling the time-series in the range $[p, q]$ with $-1 \leq p < q \leq 1$ by

$$\hat{x}_i = p + (q - p) \cdot \frac{x_i - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})}, \forall i \in \{1, \dots, n\}, \quad (4)$$

and computes the cosine of the sum of the angles for the Gramian angular summation field [112] by

$$\text{GASF}_{i,j} = \cos(\phi_i + \phi_j), \forall i, j \in 1, \dots, n, \quad (5)$$

with $\phi_i = \arccos(\hat{x}_i), \forall i \in \{1, \dots, n\}$ being the polar coordinates. We generate image datasets based on signal data with different noise parameters ($b \in \{0.0, \dots, 1.95\}$) to show the influence of the image data on the classification accuracy. As an example, Figure 11 shows the Gramian angular summation field plots for the noise parameters $b = [0, 0.5, 1.0, 1.5, 1.95]$. We present the Gramian angular summation field for the classes 0, 5, and 9 to show the dependency of the frequency of the signal data on the Gramian angular summation field.

Models. We use the following models for classification. Our encoder for time-series classification consists of a 1D convolutional layer (filter size 50, kernel 4), a max pooling layer (pool size 4), batch normalization, and a dropout layer (20%). The image encoder consists of a layer normalization and 2D convolutional layer (filter size 200), and batch normalization with ELU activation. After that, we add a 1D convolutional layer (filter size 200, kernel 4), max pooling (pool size 2), batch normalization, and 20% dropout. For both models, after the dropout layer follows a cross-modal representation – i.e., an LSTM with 10 units, a Dense layer with 20 units, a batch normalization

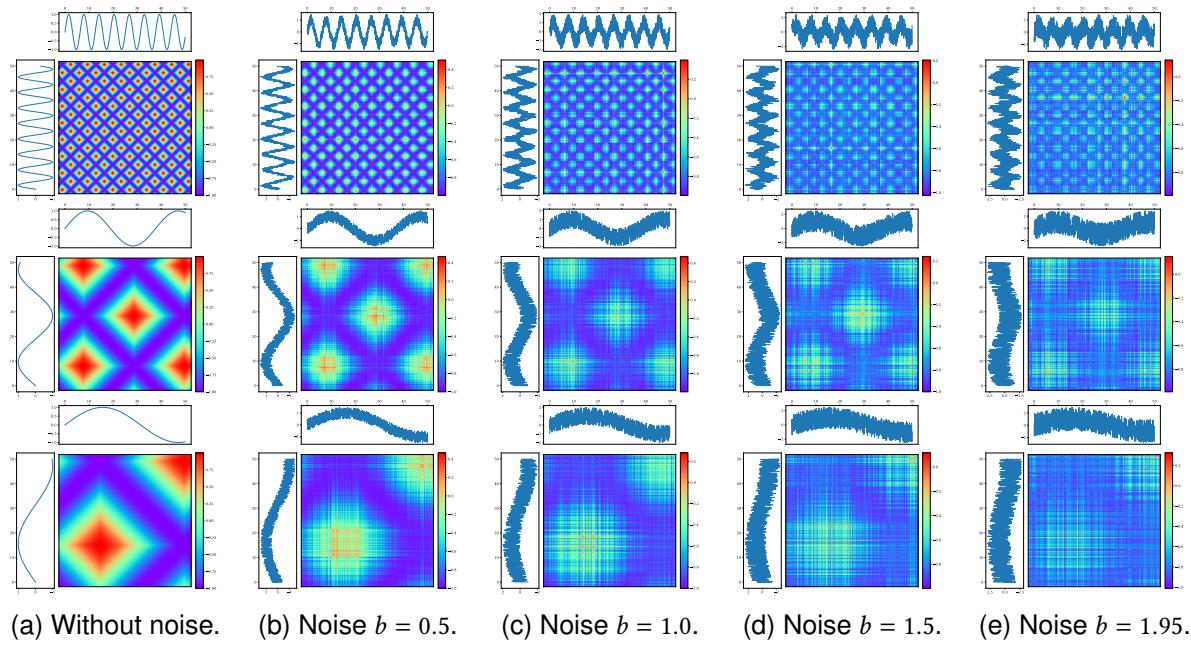


Fig. 11. Plot of the Gramian angular summation field based on 1D signal data with added noise for the classes 0 (top row), 5 (middle row) and 9 (bottom row).

layer, and a Dense layer of 10 units (for 10 sinusoidal classes). These layers are shared between both models.

A.6 Details on Architectures for Offline HWR

In this section, we provide details about the integration of Inception [101], ResNet [51] and gated text recognizer [121] modules into the offline HWR system. All three architectures are based on publicly available implementations, but we changed or adapted the first layer for the image input and the last layer for a proper input for our latent representation module.

Inception. Figure 12 gives an overview of the integration of the Inception module. The Inception module is part of the well-known GoogLeNet architecture. The main idea is to consider how an optimal local sparse structure can be approximated by readily available dense components. As the merging of pooling layer outputs with convolutional layer outputs would lead to an inevitable increase in the number of output and would lead to a high computational increase, we apply the Inception module with dimensionality reduction to our offline HWR approach [101]. The input image is of size $H \times W$. What follows is the Inception (3a), Inception (3b), a max pooling layer (3×3) and Inception (4a). We add three 1D convolutional layers to obtain an output dimensionality of 400×200 as the input for the latent representation.

ResNet34. Figure 13 provides an overview of the integration of the ResNet34 architecture. Instead of learning unreference functions, [51] reformulated the layers as learning residual functions with reference to the layer inputs. This residual network is easier to optimize and can gain accuracy from considerably increased depth. The ResNet block allows the layers to fit a residual mapping denoted as $\mathcal{H}(\mathbf{x})$ with identity \mathbf{x} and fits the mapping $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$. The original mapping is recast into $\mathcal{F}(\mathbf{x}) + \mathbf{x}$. We reshape the output of ResNet34, add a 1D convolutional layer, and reshape the output for the latent representation.

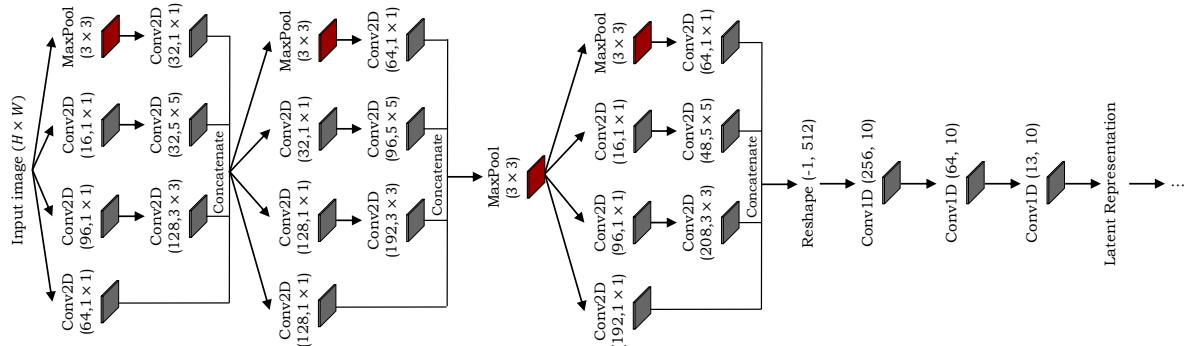


Fig. 12. Offline HWR method based on Inception modules [101].

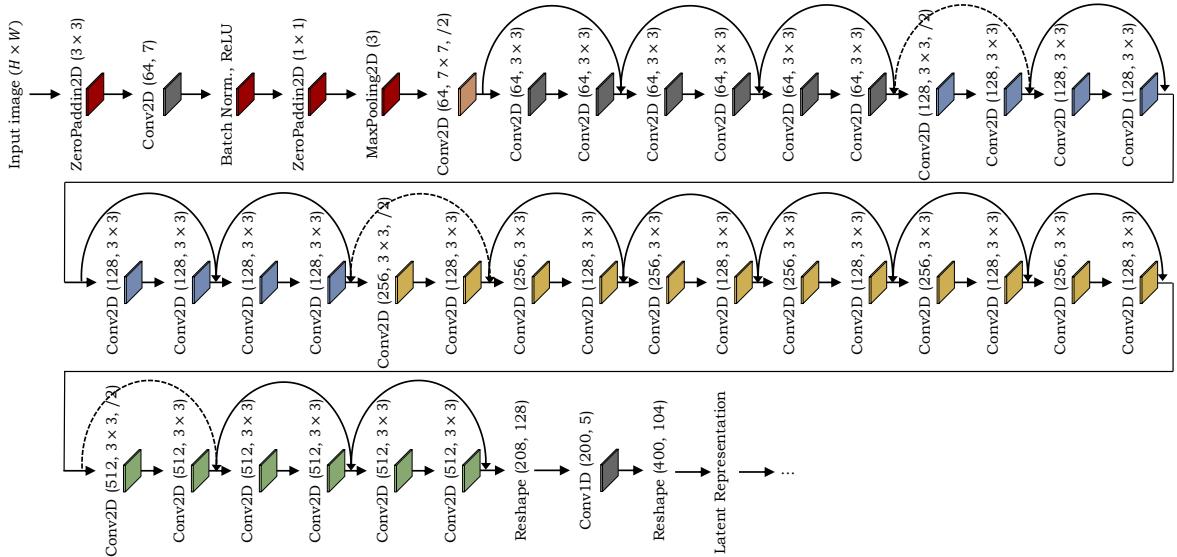


Fig. 13. Offline HWR method based on the ResNet34 architecture [51].

Gated Text Recognizer. Figure 14 gives an overview of the integration of the gated text recognizer [121] module – a fully convolutional network that uses batch normalization and layer normalization to regularize the training process and increase convergence speed. The module uses batch renormalization [56] on all batch normalization layers. Depthwise separable convolutions reduce the number of parameters at the same/better classification performance. The gated text recognizer uses spatial dropout instead of regular unstructured dropout for better regularization. After the input image of size $H \times W$ that is normalized follows a convolutional layer with Softmax normalization, a 13×13 filter, and dropout (40%). After the dropout layer, a stack of 2, 4, 6 or 8 gate blocks follows that models the input sequence. Similar to [121], we add a dropout of 20% after the last gated text recognizer block. Lastly, we add a 2D convolutional layer of 200, a batch normalization layer and a layer normalization layer that is the input for our latent representation.

A.7 Detailed Online HWR Evaluation

Table 5 gives an overview of cross-modal representation learning results based on two convolutional layers ($c = 2$) for the cross-modal representation. Our CNN+BiLSTM contains three additional convolutional layers and outperforms the smaller CNN+BiLSTM by [80] on the WD classification

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss

256

Functions for Online Handwriting Recognition

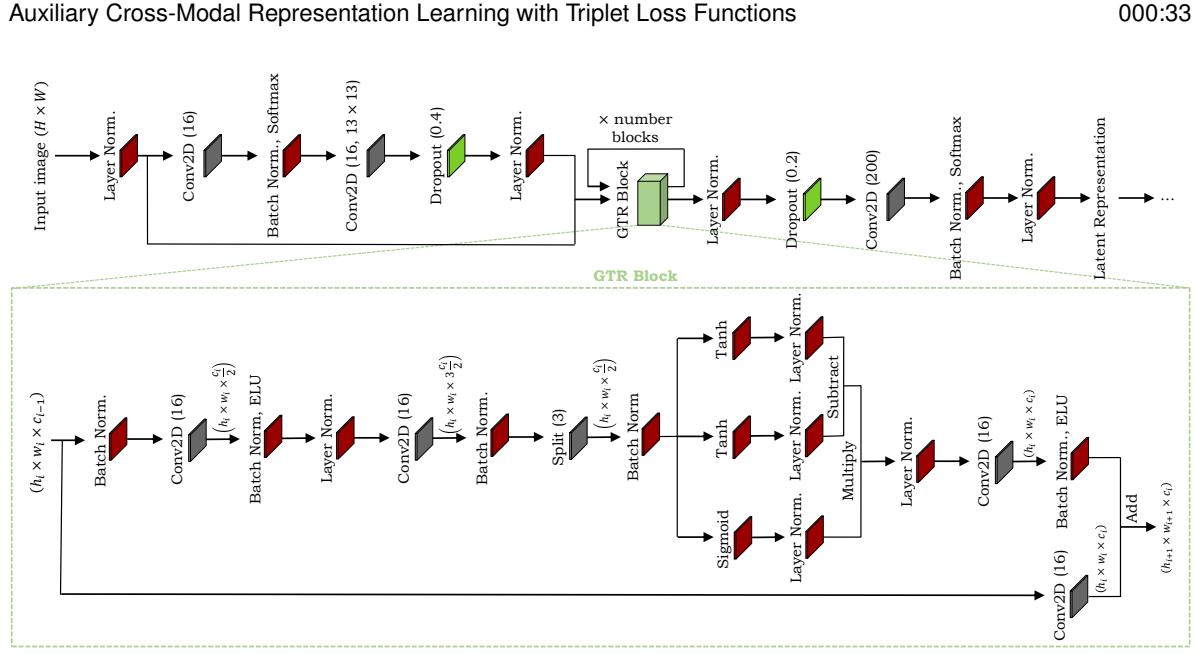


Fig. 14. Offline HWR method based on the gated text recognizer architecture [121].

Table 10. Evaluation results (WER and CER in %) averaged over five splits of the baseline time-series-only technique and our cross-modal learning technique for the inertial-based OnHW datasets [80] with and without mutated vowels (MV) for two convolutional layers $c = 2$. We propose writer-(in)dependent (WD/WI) results. Best results are **bold**, and second best results are underlined. Arrows indicate improvements (\uparrow) and degradation (\downarrow) of baseline results (CNN+BiLSTM, w/o MV).

Method	OnHW-words500				OnHW-wordsRandom			
	WD		WI		WD		WI	
	WER	CER	WER	CER	WER	CER	WER	CER
Small CNN+BiLSTM, \mathcal{L}_{CTC} , w/ MV	51.95	17.16	60.91	27.80	41.27	7.87	84.52	35.22
CNN+BiLSTM (ours), \mathcal{L}_{CTC} , w/ MV	42.81	13.04	60.47	28.30	37.13	6.75	83.28	35.90
CNN+BiLSTM (ours), \mathcal{L}_{CTC} , w/o MV	42.77	13.44	59.82	28.54	41.52	7.81	83.54	36.51
\mathcal{L}_{MSE}	39.79 \uparrow	12.14 \uparrow	60.35 \downarrow	28.48 \uparrow	39.98 \uparrow	7.79 \uparrow	83.50 \uparrow	36.92 \downarrow
\mathcal{L}_{CS}	43.40 \downarrow	13.70 \downarrow	59.31 \uparrow	27.99 \uparrow	40.31 \uparrow	7.68 \uparrow	83.68 \downarrow	36.30 \uparrow
\mathcal{L}_{PC}	38.90 \uparrow	<u>11.60</u> \uparrow	60.77 \downarrow	28.45 \uparrow	39.93 \uparrow	7.60 \uparrow	83.19 \uparrow	35.83 \uparrow
\mathcal{L}_{KL}	37.25 \uparrow	11.29 \uparrow	65.10 \downarrow	31.26 \downarrow	41.81 \downarrow	8.22 \downarrow	84.40 \downarrow	38.93 \downarrow
$\mathcal{L}_{trpl,2}(\mathcal{L}_{MSE})$	41.16 \uparrow	12.71 \uparrow	<u>58.65</u> \uparrow	28.19 \uparrow	41.16 \uparrow	8.03 \downarrow	85.38 \downarrow	39.49 \downarrow
$\mathcal{L}_{trpl,2}(\mathcal{L}_{CS})$	42.74 \uparrow	13.43 \uparrow	58.13 \uparrow	27.62 \uparrow	41.49 \uparrow	8.18 \downarrow	85.24 \downarrow	38.75 \downarrow
$\mathcal{L}_{trpl,2}(\mathcal{L}_{PC})$	39.94 \uparrow	12.19 \uparrow	62.76 \downarrow	30.68 \downarrow	41.58 \downarrow	8.18 \downarrow	85.18 \downarrow	38.53 \downarrow
$\mathcal{L}_{trpl,2}(\mathcal{L}_{KL})$	<u>38.34</u> \uparrow	11.77 \uparrow	67.08 \downarrow	33.84 \downarrow	41.87 \downarrow	8.33 \downarrow	86.34 \downarrow	40.37 \downarrow

tasks. Without triplet loss, \mathcal{L}_{PC} yields the best results on the OnHW-wordsRandom dataset. The triplet loss partly decreases results and partly improves results on the OnHW-words500 dataset. In conclusion, two convolutional layers for the cross-modal representation has a negative impact, while here the triplet loss has no impact.

REFERENCES

[1] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. 2021. VATT: Transformers for Multimodal Self-Supervised Learning from Raw Video, Audio and Text. In *Advances in Neural Information Processing Systems (NIPS)*.

[2] Emre Aksan, Fabrizio Pece, and Otmar Hilliges. 2018. DeepWriting: Making Digital Ink Editable via Deep Generative Modeling. In *CHI Conf. on Human Factors in Computing Systems*. 1–14. <https://doi.org/10.1145/3173574.3173779>

[3] Eloi Alonso, Bastien Moysset, and Ronaldo Messina. 2019. Adversarial Generation of Handwritten Text Images Conditioned on Sequences. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Sydney, Australia. <https://doi.org/10.1109/ICDAR.2019.00083>

[4] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. 2022. data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 162. 1298–1312.

[5] Grigoris Bastas, Kosmas Kritsis, and Vassilis Katsouros. 2020. Air-Writing Recognition using Deep Convolutional and Recurrent Neural Network Architectures. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Dortmund, Germany, 7–12. <https://doi.org/10.1109/ICFHR2020.2020.00013>

[6] Roman Bertolami and Horst Bunke. 2018. Hidden Markov Model-based Ensemble Methods for Offline Handwritten Text Line Recognition. In *Pattern Recognition*, Vol. 41(11). 3452–3460. <https://doi.org/10.1016/j.patcog.2008.04.003>

[7] Ayan Kumar Bhunia, Abhirup Das, Ankan Kumar Bhunia, Perla Sai Raj Kishore, and Partha Pratim Roy. 2019. Handwriting Recognition in Low-Resource Scripts Using Adversarial Learning. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA. <https://doi.org/10.1109/CVPR.2019.00490>

[8] Ali Furkan Biten, Andrés Mafla, Lluís Gómez, and Dimosthenis Karatzas. 2022. Is an Image Worth Five Sentences? A New Look into Semantics for Image-Text Matching. In *IEEE/CVF Winter Conf. on Applications of Computer Vision (WACV)*. Waikoloa, HI. <https://doi.org/10.1109/WACV51458.2022.00254>

[9] Théodore Bluche. 2015. Deep Neural Networks for Large Vocabulary Handwritten Text Recognition. In *Thèse de Doctorat*. Université Paris-Sud.

[10] Théodore Bluche. 2016. Joint Line Segmentation and Transcription for End-to-End Handwritten Paragraph Recognition. In *Advances in Neural Information Processing Systems (NIPS)*. 838–846. <https://doi.org/10.5555/3157096.3157190>

[11] Théodore Bluche, Jérôme Louradour, and Ronaldo Messina. 2017. Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan. <https://doi.org/10.1109/ICDAR.2017.174>

[12] Théodore Bluche and Ronaldo Messina. 2017. Gated Convolutional Recurrent Neural Networks for Multilingual Handwriting Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan, 646–651. <https://doi.org/10.1109/ICDAR.2017.111>

[13] Hervé Bredin. 2017. TristouNet: Triplet Loss for Speaker Turn Embedding. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. 5430–5434. <https://doi.org/10.1109/ICASSP.2017.7953194>

[14] Manuel Carbonell, Joan Mas, Mauricio Villegas, Alicia Fornés, and Josep Lladós. 2019. End-to-End Handwritten Text Detection and Transcription in Full Pages. In *Intl. Conf. on Document Analysis and Recognition Workshops (ICDARW)*. Sydney, NSW. <https://doi.org/10.1109/ICDARW.2019.40077>

[15] Victor Carbune, Pedro Gonnet, Thomas Deselaers, Henry A. Rowley, Alexander Daryin, Marcos Calvo, Li-Lun Wang, Daniel Keysers, Sandro Feuz, and Philippe Gervais. 2020. Fast Multi-language LSTM-based Online Handwriting Recognition. In *Intl. Journal on Document Analysis and Recognition (IJDAR)*, Vol. 23. 89–102. <https://doi.org/10.1007/s10032-020-00350-4>

[16] Dayvid Castro, Byron L. D. Bezerra, and Mêuser Valença. 2018. Boosting the Deep Multidimensional Long-Short-Term Memory Network for Handwritten Recognition Systems. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Niagara Falls, NY. <https://doi.org/10.1109/ICFHR-2018.2018.00031>

[17] Ushasi Chaudhuri, Biplab Banerjee, Avik Bhattacharya, and Mihai Datcu. 2020. CrossATNet - A Novel Cross-Attention based Framework for Sketch-based Image Retrieval. In *Image and Vision Computing*, Vol. 104. <https://doi.org/10.1016/j.imavis.2020.104003>

[18] Brian Chen, Andrew Rouditchenko, Kevin Duarte, Hilde Kuehne, Samuel Thomas, Angie Boggust, Rameswar Panda, Brian Kingsbury, Rogerio Feris, David Harwath, James Glass, Michael Picheny, and Shih-Fu Chang. 2021. Multimodal Clustering Networks for Self-supervised Learning from Unlabeled Videos. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Montreal, QC. <https://doi.org/10.1109/ICCV48922.2021.00791>

[19] Dapeng Chen, Min Wang, Haobin Chen, Lin Wu, Jing Qin, and Wei Peng. 2022. Cross-Modal Retrieval with Heterogeneous Graph Embedding. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 3291–3300. <https://doi.org/10.1145/3503161.3548195>

[20] Jiacheng Chen, Hexiang Hu, Hao Wu, Yuning Jiang, and CHanghu Wang. 2022. Learning the Best Pooling Strategy for Visual Semantic Embedding. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN. <https://doi.org/10.1109/CVPR46437.2021.01553>

[21] Tianlang Chen, Jiajun Deng, and Jiebo Luo. 2020. Adaptive Offline Quintuplet Loss for Image-Text Matching. In *Europ. Conf. on Computer Vision (ECCV)*.

[22] Zhuo Chen, Yichao Wu, Fei Yin, and Cheng-Lin Liu. 2017. Simultaneous Script Identification and Handwriting Recognition via Multi-Task Learning of Recurrent Neural Networks. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan, 525–530. <https://doi.org/10.1109/ICDAR.2017.92>

[23] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a Similarity Metric Discriminatively, with Application to Face Verification. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. San Diego, CA. <https://doi.org/10.1109/CVPR.2005.202>

[24] Arindam Chowdhury and Lovekesh Vig. 2018. An Efficient End-to-End Neural Model for Handwritten Text Recognition. In *British Machine Vision Conference (BMVC)*.

[25] Sanghyuk Chun, Seong Joon Oh, Rafael Sampaio de Rezende, Yannis Kalantidis, and Diane Larlus. 2021. Probabilistic Embeddings for Cross-Modal Retrieval. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN. <https://doi.org/10.1109/CVPR46437.2021.00831>

[26] Jonathan Chung and Thomas Delteil. 2019. A Computationally Efficient Pipeline Approach to Full Page Offline Handwritten Text Recognition. In *Intl. Conf. on Document Analysis and Recognition Workshops (ICDARW)*. Sydney, NSW. <https://doi.org/10.1109/ICDARW.2019.40078>

[27] Shohreh Deldari, Hao Xue, Aaqib Saeed, Daniel V. Smith, and Flora D. Salim. 2022. COCOA: Cross Modality Contrastive Learning for Sensor Data. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, Vol. 6(3), article 108. 1–28. <https://doi.org/10.1145/3550316>

[28] Cheng Deng, Zhaojia Chen, Xianglong Liu, Xinbo Gao, and Dacheng Tao. 2018. Triplet-based Deep Hashing Network for Cross-Modal Retrieval. In *IEEE Trans. on Image Processing*, Vol. 27(8). 32893–3903. <https://doi.org/10.1109/TIP.2018.2821921>

[29] Haiwen Diao, Ying Zhang, Lin Ma, and Huchuan Lu. 2021. Similarity Reasoning and Filtration for Image-Text Matching. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 35(2). 1218–1226. <https://doi.org/10.1609/aaai.v35i2.16209>

[30] Thanh-Toan Do, Toan Tran, Ian Reid, Vijay Kumar, Tuan Hoang, and Gustavo Carneiro. 2019. A Theoretically Sound Upper Bound on the Triplet Loss for Improving the Efficiency of Deep Distance Metric Learning. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, 10404–10413. <https://doi.org/10.1109/CVPR.2019.01065>

[31] Patrick Doetsch, Michal Koziełski, and Hermann Ney. 2014. Fast and Robust Training of Recurrent Neural Networks for Offline Handwriting Recognition. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Hersonissos, Greece. <https://doi.org/10.1109/ICFHR.2014.54>

[32] Philippe Drewu, Patrick Doetsch, Christian Plahl, and Hermann Ney. 2011. Hierarchical Hybrid MLP/HMM or Rather MLP Features for a Discriminatively Trained Gaussian HMM: A Comparison for Offline Handwriting Recognition. In *IEEE Intl. Conf. on Image Processing (ICIP)*. Brussels, Belgium. <https://doi.org/10.1109/ICIP.2011.6116480>

[33] Kartik Dutta, Praveen Krishnan, Minesh Mathew, and C. V. Jawahar. 2018. Improving CNN-RNN Hybrid Networks for Handwriting Recognition. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Niagara Falls, NY, 80–85. <https://doi.org/10.1109/ICFHR-2018.2018.00023>

[34] S. España-Boquera, M.J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez. 2011. Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 33(4). 767–779. <https://doi.org/10.1109/TPAMI.2010.141>

[35] Fartash Faghri, David J. Fleet, Hamie Ryan Kiros, and Sanja Fidler. 2018. VSE++: Improving Visual-Semantic Embeddings with Hard Negatives. In *British Machine Vision Conf. (BMVC)*.

[36] Alex Falcon, Giuseppe Serra, and Oswald Lanz. 2022. A Feature-space Multimodal Data Augmentation Technique for Text-Video Retrieval. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 4385–4394. <https://doi.org/10.1145/3503161.3548365>

[37] Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, and Roee Litman. 2020. ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, 4324–4333. <https://doi.org/10.1109/CVPR42600.2020.00438>

[38] Andrea Frome, Greg S. Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc’ Aurelio Ranzato, and Tomas Mikolov. 2013. DeViSE: A Deep Visual-Semantic Embedding Model. In *Advances in Neural Information Processing Systems (NIPS)*.

[39] Albert Gordo and Diane Larlus. 2017. Beyond Instance-Level Image Retrieval: Leveraging Captions to Learn a Global Visual Representation for Semantic Retrieval. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI. <https://doi.org/10.1109/CVPR.2017.560>

[40] Alex Graves. 2014. Generating Sequences with Recurrent Neural Networks. In *arXiv preprint arXiv:1308.0850*.

[41] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. 2009. A Novel Connectionist System for Unconstrained Handwriting Recognition. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 31(5). 855–868. <https://doi.org/10.1109/TPAMI.2008.137>

[42] Alex Graves and Jürgen Schmidhuber. 2008. Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks. In *Advances in Neural Information Processing Systems (NIPS)*. 545–552. <https://doi.org/10.5555/2981780.2981848>

[43] Emmanuele Grosicki and Haikal El-Abed. 2011. ICDAR 2011 - French Handwriting Recognition Competition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Beijing, China. <https://doi.org/10.1109/ICDAR.2011.290>

[44] Jiuxiang Gu, Jianfei Cai, Shafiq Joty, Li Niu, and Gang Wang. 2018. Look, Imagine and Match: Improving Textual-Visual Cross-Modal Retrieval with Generative Models. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT. <https://doi.org/10.1109/CVPR.2018.00750>

[45] Xiangming Gu, Longshen Ou, Danielle Ong, and Ye Wang. 2022. MM-ALT: A Multimodal Automatic Lyric Transcription System. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 3328–3337. <https://doi.org/10.1145/3503161.3548411>

[46] Dan Guo, Shengeng Tang, and Meng Wang. 2019. Connectionist Temporal Modeling of Video and Language: A Joint Model for Translation and Sign Labeling. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*. 751–757. <https://doi.org/10.24963/ijcai.2019/106>

[47] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel. 2022. AudioCLIP: Extending Clip to Image, Text and Audio. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. Singapore, Singapore. <https://doi.org/10.1109/ICASSP43922.2022.9747631>

[48] Frank M. Hafner, Amran Bhuyian, Julian F. P. Kooij, and Eric Granger. 2022. Cross-Modal Distillation for RGB-Depth Person Re-Identification. In *Computer Vision and Image Understanding (CVIU)*, Vol. 103352. <https://doi.org/10.1016/j.cviu.2021.103352>

[49] Tom S. F. Haines, Oisin Mac Aodha, and Gabriel J. Brostow. 2016. My Text in Your Handwriting. In *ACM Trans. on Graphics*, Vol. 35(3). 1–18. <https://doi.org/10.1145/2886099>

[50] Ben Harwood, Vijay Kumar B.G., Gustavo Carneiro, Ian Reid, and Tom Drummond. 2017. Smart Mining for Deep Metric Learning. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Venice, Italy. <https://doi.org/10.1109/ICCV.2017.307>

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV. <https://doi.org/10.1109/CVPR.2016.90>

[52] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2020. MHTN: Modal-Adversarial Hybrid Transfer Network for Cross-Modal Retrieval. In *Trans. on Cybernetics*, Vol. 50(3). 1047–1059. <https://doi.org/10.1109/TCYB.2018.2879846>

[53] Yan Huang, Qi Wu, Chunfeng Song, and Liang Wang. 2018. Learning Semantic Concepts and Order for Image and Sentence Matching. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT. <https://doi.org/10.1109/CVPR.2018.00645>

[54] Raashid Hussain, Ahsen Raza, Imran Siddiqi, Khurram Khurshid, and Chawki Djeddi. 2015. A Comprehensive Survey of Handwritten Document Benchmarks: Structure, Usage and Evaluation. In *EURASIP Journal on Image and Video Processing*, Vol. 46. <https://doi.org/10.1186/s13640-015-0102-5>

[55] R. Reeve Ingle, Yasuhisa Fujii, Thomas Deselaers, Jonathan Baccash, and Ashok C. Popat. 2019. A Scalable Handwritten Text Recognition System. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Sydney, Australia. <https://doi.org/10.1109/ICDAR.2019.00013>

[56] Sergey Ioffe. 2017. Batch Renormalization: Towards Reducing Minibatch Dependence in Batch-Normalized Models. In *Advances in Neural Information Processing Systems (NIPS)*. 1942–1950. <https://doi.org/10.5555/3294771.3294956>

[57] Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. 2022. ColloSSL: Collaborative Self-Supervised Learning for Human Activity Recognition. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*.

[58] Bo Ji and Tianyi Chen. 2020. Generative Adversarial Network for Handwritten Text. In *arXiv preprint arXiv:1907.11845*.

[59] Lei Kang, Pau Riba, Marcal Rusinol, Alicia Fornes, and Mauricio Villegas. 2022. Pay Attention to What You Read: Non-recurrent Handwritten Text-Line Recognition. In *Pattern Recognition*, Vol. 129. <https://doi.org/10.1016/j.patcog.2022.108766>

[60] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. 2020. Proxy Anchor Loss for Deep Metric Learning. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 3238–3247.

[61] Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. 2014. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models. In *arXiv preprint arXiv:1411.2539*.

[62] Michal Kozielski, Patrick Doetsch, and Hermann Ney. 2013. Improvements in RWTH’s System for Off-Line Handwriting Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Washington, DC. <https://doi.org/10.1109/ICDAR.2013.190>

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition

260

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions

000:37

- [63] Praveen Krishnan, Kartik Dutta, and C. V. Jawahar. 2018. Word Spotting and Recognition using Deep Embedding. In *IAPR Intl. Workshop on Document Analysis Systems (DAS)*. Vienna, Austria, 1–6. <https://doi.org/10.1109/DAS.2018.70>
- [64] Hyodong Lee, Joonseok Lee, Joe Yue-Hei Ng, and Paul Natsev. 2020. Large Scale Video Representation Learning via Relational Graph Clustering. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA. <https://doi.org/10.1109/CVPR42600.2020.00684>
- [65] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. 2018. Stacked Cross Attention for Image-Text Matching. In *Europ. Conf. on Computer Vision (ECCV)*, Vol. 11208. https://doi.org/10.1007/978-3-030-01225-0_13
- [66] Kunpeng Li, Yulun Zhang, Kai Li, Yuanyuan Li, and Yun Fu. 2019. Visual Semantic Reasoning for Image-Text Matching. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Seoul, Korea. <https://doi.org/10.1109/ICCV.2019.00475>
- [67] Nan Li, Jinying Chen, Huagu Cao, Bing Zhang, and Prem Natarajan. 2014. Applications of Recurrent Neural Network Language Model in Offline Handwriting Recognition and Word Spotting. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Hersonissos, Greece. <https://doi.org/10.1109/ICFHR.2014.30>
- [68] Wenbin Li, Xuesong Yang, Meihao Kong, Lei Wang, Jing Huo, Yang Gao, and Jiebo Luo. 2021. Triplet is All You Need with Random Mappings for Unsupervised Visual Representation Learning. In *arXiv preprint arXiv:2107.10419*.
- [69] Dongyun Liang, Weiran Xu, and Ying Zhao. 2017. Combining Word-Level and Character-Level Representations for Relation Classification of Informal Text (RepL4NLP). In *Workshop on Representation Learning for NLP (RepL4NLP)*. Vancouver, Canada, 43–47. <https://doi.org/10.18653/v1/W17-2606>
- [70] Jae Hyun Lim, Pedro O. O. Pinheiro, Negar Rostamzadeh, Chris Pal, and Sungjin Ahn. 2019. Neural Multisensory Scene Inference. In *Advances in Neural Information Processing Systems (NIPS)*, Vol. 32(807). 8996–9006. <https://doi.org/10.5555/3454287.3455094>
- [71] Hui Lin, Zhiheng Ma, Xiaopeng Hong, Yaowei Wang, and Zhou Su. 2022. Semi-supervised Crowd Counting via Density Agency. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 1416–1426. <https://doi.org/10.1145/3503161.3547867>
- [72] Shikun Liu, Edward Johns, and Andrew J. Davison. 2019. End-to-End Multi-Task Learning with Attention. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, 1871–1880. <https://doi.org/10.1109/CVPR.2019.00197>
- [73] Marcus Liwicki and Horst Bunke. 2005. IAM-OnDB - an On-Line English Sentence Database Acquired from Handwritten Text on a Whiteboard. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Seoul, Korea, 956–961. <https://doi.org/10.1109/ICDAR.2005.132>
- [74] Mingsheng Long, Yue Cao, Lianmin Wang, and Michael I. Jordan. 2015. Learning Transferable Features with Deep Adaptation Networks. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 37. 97–105. <https://doi.org/10.5555/3045118.3045130>
- [75] Farès Menasri, Jérôme Louradour, Anne-Laure Bianne-Bernard, and Christopher Kermorvant. 2012. The A2iA French Handwriting Recognition System at the RIMES-ICDAR2011 Competition. In *Proc. of the Intl. Society for Optical Engineering (SPIE)*, Vol. 8297(51). <https://doi.org/10.1117/12.911981>
- [76] Ronaldo Messina and Christopher Kermorvant. 2014. Over-Generative Finite State Transducer N-Gram for Out-of-Vocabulary Word Recognition. In *IAPR Intl. Workshop on Document Analysis Systems*. Tours, France. <https://doi.org/10.1109/DAS.2014.24>
- [77] Johannes Michael, Roger Labahn, Tobias Grüning, and Jochen Zöllner. 2019. Evaluating Sequence-to-Sequence Models for Handwritten Text Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Sydney, NSW. <https://doi.org/10.1109/ICDAR.2019.00208>
- [78] Yasunori Ohishi, Marc Delcroix, Tsubasa Ochiai, Shoko Araki, Daiki Takeuchi, Daisuke Niizumi, Akisato Kimura, Noboru Harada, and Kunio Kashino. 2022. ConceptBeam: Concept Driven Target Speech Extraction. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 4252–4260. <https://doi.org/10.1145/3503161.3548397>
- [79] Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. 2022. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. Lisboa, Portugal, 5934–5943. <https://doi.org/10.1145/3503161.3548167>
- [80] Felix Ott, David Rügamer, Lucas Heublein, Tim Hamann, Jens Barth, Bernd Bischl, and Christopher Mutschler. 2022. Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In *International Journal on Document Analysis and Recognition (IJDAR)*, Vol. 25(12). 385–414. <https://doi.org/10.1007/s10032-022-00415-6>
- [81] Joan Pastor-Pellicer, Salvador Espa na Boquera, M. J. Castro-Bleda, and Francisco Zamora-Martínez. 2015. A Combined Convolutional Neural Network and Dynamic Programming Approach for Text Line Normalization. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Tunis, Tunisia, 341–345. <https://doi.org/10.1109/ICDAR.2015.7333780>

[82] Yuxin Peng and Jinwei Qi. 2019. CM-GANs: Cross-Modal Generative Adversarial Networks for Common Representation Learning. In *ACM Trans. on Multimedia Computing, Communications, and Applications*, Vol. 15(1). 1–24. <https://doi.org/10.1145/3284750>

[83] Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. 2014. Dropout Improves Recurrent Neural Networks for Handwriting Recognition. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Hersonissos, Greece, 285–290. <https://doi.org/10.1109/ICFHR.2014.55>

[84] AJ Piergiovanni, Anelia Angelova, and Michael S. Ryoo. 2022. Evolving Losses for Unsupervised Video Representation Learning. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA. <https://doi.org/10.1109/CVPR42600.2020.00021>

[85] Rejean Plamondon and Sargur N. Srihari. 2000. On-line and Off-line Handwriting Recognition: A Comprehensive Survey. In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 22(1). 63–84. <https://doi.org/10.1109/34.824821>

[86] Arik Poznanski and Lior Wolf. 2016. CNN-N-Gram for Handwriting Word Recognition. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, 2306–2314. <https://doi.org/10.1109/CVPR.2016.253>

[87] Joan Puigcerver. 2017. Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan, 67–72. <https://doi.org/10.1109/ICDAR.2017.20>

[88] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Intl. Conf. on Machine Learning (ICML)*, Vol. 139. 8748–8763.

[89] Viresh Ranjan, Nikhil Rasiwasia, and C. V. Jawahar. 2015. Multi-Label Cross-Modal Retrieval. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Santiago de Chile, Chile. <https://doi.org/10.1109/ICCV.2015.466>

[90] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Covello, Gabriel Doyle, Gert R.G. Lanckriet, Roger Levy, and Nuno Vasconcelos. 2010. A New Approach to Cross-Modal Multimedia Retrieval. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 251–260. <https://doi.org/10.1145/1873951.1873987>

[91] Nikolaos Sarafianos, Xiang Xu, and Ioannis A. Kakadiaris. 2019. Adversarial Representation Learning for Text-to-Image Matching. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Seoul, Korea, 5814–5824. <https://doi.org/10.1109/ICCV.2019.00591>

[92] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA. <https://doi.org/10.1109/CVPR.2015.7298682>

[93] David Semedo and João Magalhães. 2020. Adaptive Temporal Triplet-loss for Cross-modal Embedding Learning. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 1152–1161. <https://doi.org/10.1145/3394171.3413540>

[94] Annapurna Sharma, Rahul Ambati, and Dinesh Babu Jayagopi. 2020. Towards Faster Offline Handwriting Recognition using Temporal Convolutional Networks. In *NCVPRIPG 2019 Communications in Computer and Information Science (CCIS)*, Springer, Singapore, Vol. 1249. 344–354. https://doi.org/10.1007/978-981-15-8697-2_32

[95] Annapurna Sharma and Dinesh Babu Jayagopi. 2021. Towards Efficient Unconstrained Handwriting Recognition using Dilated Temporal Convolutional Network. In *Expert Systems with Applications*, Vol. 164. <https://doi.org/10.1016/j.eswa.2020.114004>

[96] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. 2022. FLAVA: A Foundational Language and Vision Alignment Model. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, 15638–15650. <https://doi.org/10.1109/CVPR52688.2022.01519>

[97] Sumeet S. Singh and Sergey Karayev. 2021. Full Page Handwriting Recognition via Image to Sequence Extraction. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Lausanne, Switzerland, 55–69. https://doi.org/10.1007/978-3-030-86334-0_4

[98] Yale Song and Mohammad Soleymani. 2019. Polysemous Visual-Semantic Embedding for Cross-Modal Retrieval. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA. <https://doi.org/10.1109/CVPR.2019.00208>

[99] Sebastian Sudholt and Gernot A. Fink. 2018. Attribute CNNs for Word Spotting in Handwritten Documents. In *Intl. Journal on Document Analysis and Recognition (IJDAR)*, Vol. 21. 199–218. <https://doi.org/10.1007/s10032-018-0295-0>

[100] Jorge Sueiras, Victoria Ruiz, Angel Sanchez, and Jose F. Velez. 2018. Offline Continuous Handwriting Recognition Using Sequence-to-Sequence Neural Networks. In *Neurocomputing*, Vol. 289(C). 119–128. <https://doi.org/10.1016/j.neucom.2018.02.008>

[101] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going Deeper with Convolutions. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, 1–9. <https://doi.org/10.1109/CVPR.2015.7298594>

8. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition

262

Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions

000:39

- [102] Bing Tian, Yong Zhang, Jin Wang, and Chunxiao Xing. 2019. Hierarchical Inter-Attention Network for Document Classification with Multi-Task Learning. In *Intl. Joint Conf. on Artificial Intelligence (IJCAI)*. 3569–3575. <https://doi.org/10.24963/ijcai.2019/495>
- [103] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation Learning with Contrastive Predictive Coding. In *arXiv preprint arXiv:1807.03748*.
- [104] Shashanka Venkataraman, Ewa Kijak, Laurent Amsaleg, and Yannis Avrithis. 2022. AlignMix: Improving Representations by Interpolating Aligned Features. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 19174–19183.
- [105] Paul Voigtlaender, Patrick Doetsch, and Hermann Ney. 2016. Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks. In *Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*. Shenzhen, China, 228–233. <https://doi.org/10.1109/ICFHR.2016.0052>
- [106] Paul Voigtlaender, Patrick Doetsch, Simon Wiesler, Ralf Schlüter, and Hermann Ney. 2015. Sequence-Discriminative Training of Recurrent Neural Networks. In *Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*. Brisbane, QLD. <https://doi.org/10.1109/ICASSP.2015.7178341>
- [107] Qian Wan and Qin Zou. 2021. Learning Metric Features for Writer-Independent Signature Verification using Dual Triplet Loss. In *Intl. Conf. on Pattern Recognition (ICPR)*. Milan, Italy, 3853–3859. <https://doi.org/10.1109/ICPR48806.2021.9413091>
- [108] Haoran Wang, Ying Zhang, Zhong Ji, Yanwei Pang, and Lin Ma. 2020. Consensus-Aware Visual-Semantic Embedding for Image-Text Matching. In *Europ. Conf. on Computer Vision (ECCV)*.
- [109] Junsheng Wang, Tiantian Gong, Zhixiong Zeng, Changchang Sun, and Yan Yan. 2022. C³CMR: Cross-Modality Cross-Instance Contrastive Learning for Cross-Media Retrieval. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*. 4300–4308. <https://doi.org/10.1145/3503161.3548263>
- [110] Luyu Wang, Pauline Luc, Adrià Recasens, Jean-Baptiste Alayrac, and Aäron van den Oord. 2021. Multimodal Self-Supervised Learning of General Audio Representations. In *arXiv preprint arXiv:2104.12807*.
- [111] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Canjie Luo, Xiaoxue Chen, Yaqiang Wu, Qianying Wang, and Mingxiang Cai. 2020. Decoupled Attention Network for Text Recognition. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 34(7). 12216–12224. <https://doi.org/10.1609/aaai.v34i07.6903>
- [112] Zhiguang Wang and Tim Oates. 2015. Imaging Time-Series to Improve Classification and Imputation. In *Intl. Joint. Conf. on Artificial Intelligence (IJCAI)*. Buenos Aires, Argentinia, 3939–3945. <https://doi.org/10.5555/2832747.2832798>
- [113] Jonas Wehrmann, Mauricio Armani Lopes, Douglas Souza, and Rodrigo Barros. 2019. Language-Agnostic Visual-Semantic Embeddings. In *IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*. Seould, Korea. <https://doi.org/10.1109/ICCV.2019.00590>
- [114] Yunchao Wei, Yao Zhao, Canyi Lu, Shikui Wei, Luoqi Liu, Zhenfeng Zhu, and Shuicheng Yan. 2016. Cross-Modal Retrieval with CNN Visual Features: A New Baseline. In *Trans. on Cybernetics*, Vol. 47(2). 449–460. <https://doi.org/10.1109/TCYB.2016.2519449>
- [115] Curtis Wigington, Chris Tensmeyer, Brian Davis, William Barrett, Brian Price, and Scott Cohen. 2018. Start, Follow, Read: End-to-End Full-Page Handwriting Recognition. In *Europ. Conf. on Computer Vision (ECCV)*, Vol. 11210. 372–388. https://doi.org/10.1007/978-3-030-01231-1_23
- [116] Hao Wu, Jiayuan Mao, Yufeng Zhang, Yuning Jiang, Lei Li, Weiwei Sun, and Wei-Ying Ma. 2019. Unified Visual-Semantic Embeddings: Bridging Vision and Language With Structured Meaning Representations. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA. <https://doi.org/10.1109/CVPR.2019.00677>
- [117] Yijung Xiao and Kyunghyun Cho. 2016. Efficient Character-level Document Classification by Combining Convolution and Recurrent Layers. In *arXiv preprint arXiv:1602.00367*.
- [118] Tomoki Yoshida, Ichiro Takeuchi, and Masayuki Karasuyama. 2019. Safe Triplet Screening for Distance Metric Learning. In *Neural Computation*, Vol. 31(12). 2432–2491. https://doi.org/10.1162/neco_a_01240
- [119] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From Image Descriptions to Visual Denotations: New Similarity Metrics for Semantic Interference Over Event Descriptions. In *Trans. of the Association for Computational Linguistics (ACL)*, Vol. 2. Cambridge, MA, 67–78. https://doi.org/10.1162/tacl_a_00166
- [120] Mohamed Yousef and Tom E. Bishop. 2020. OrigamiNet: Weakly-Supervised, Segmentation-Free, One-Step, Full Page Text Recognition by Learning to Unfold. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, 14710–14719. <https://doi.org/10.1109/CVPR42600.2020.01472>
- [121] Mohamed Yousef, Khaled F. Hussain, and Osama S. Mohammed. 2020. Accurate, Data-Efficient, Unconstrained Text Recognition with Convolutional Neural Networks. In *Pattern Recognition*, Vol. 108.
- [122] Donghuo Zeng, Yi Yu, and Keizo Oyama. 2020. Deep Triplet Neural Networks with Cluster-CCA for Audio-Visual Cross-Modal Retrieval. In *ACM Trans. on Multimedia Computing, Communications, and Applications*, Vol. 16(3). 1–23. <https://doi.org/10.1145/3387164>

- [123] Xiangsheng Zeng, Donglai Xiang, Liangrui Peng, Changsong Liu, and Xiaoqing Ding. 2017. Local Discriminant Training and Global Optimization for Convolutional Neural Network Based Handwritten Chinese Character Recognition. In *IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*. Kyoto, Japan. <https://doi.org/10.1109/ICDAR.2017.70>
- [124] Ji Zhang, Yannis Kalantidis, Marcus Rohrbach, Ahmed Elgammal, and Mohamed Elhoseiny. 2019. Large-Scale Visual Relationship Understanding. In *Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 33(1). 9185–9194. <https://doi.org/10.1609/aaai.v33i01.33019185>
- [125] Liangli Zhen, Peng Hu, Xu Wang, and Dezhong Peng. 2019. Deep Supervised Cross-Modal Retrieval. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, 10394–10403. <https://doi.org/10.1109/CVPR.2019.01064>

Received 19 January 2023; revised 19 January 2023; accepted 19 January 2023

Chapter 9

Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition

Contributing Article

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition. In *IAPR International Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS)*, volume 13643, pages 373-383, Montreal, Canada, August 2022. doi:10.1007/978-3-031-37660-3_26.

Publisher Website

The final authenticated version is available online at https://link.springer.com/chapter/10.1007/978-3-031-37660-3_26.

Attached version: *arXiv preprint arXiv:2301.06293 [cs.CV]*

Copyright License

Copyright held by Owner/Author. This is the author's version of the work. It is posted here for your personal use. Not for distribution. The definite Version of Record was accepted at *IAPR Intl. Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS)*, August 2022.

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing, visualization, and project administration. David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Lucas Heublein contributed for the methodology (i.e., creation of models), software, validation, investigation, and data curation (i.e., data recording and data management). Bernd Bischl contributed by supervision. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), visualization (i.e., presentation of the published work), supervision, and funding acquisition of the project “Schreibtrainer” (grant number 16SV8228) by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”. This paper was presented by Felix Ott at the IAPR International Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS) along the International Conference on Pattern Recognition (ICPR) in August 2022.

Datasets

This paper uses the OnHW-words500, OnHW-wordsRandom, and OnHW-wordsTraj datasets.

Statement about Recent, Related Research¹⁷

As of the latest literature review, we did not find any relevant studies on DA between tablet and paper-based OnHW recognition. For a comprehensive overview of recent research on DA for time-series classification, please refer to the statement about relevant literature for the contributing paper (Ott et al., 2022a) in Chapter 7.

Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition

Felix Ott^{1,2}[0000-0002-4392-0830], David Rügamer^{2,3}[0000-0002-8772-9202],
 Lucas Heublein¹[0000-0001-6670-3698], Bernd Bischl²[0000-0001-6002-6980], and
 Christopher Mutschler¹[0000-0001-8108-0230]

¹ Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits IIS
 {felix.ott, heublein, christopher.mutschler}@iis.fraunhofer.de
² LMU Munich, Munich, Germany
 {david.ruegamer, bernd.bischl}@stat.uni-muenchen.de
³ RWTH Aachen, Aachen, Germany

Copyright held by Owner/Author. This is the author's version of the work. It is posted here for your personal use. Not for distribution. The definite Version of Record was accepted at *IAPR Intl. Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS)*, August 2022.

Abstract. The performance of a machine learning model degrades when it is applied to data from a similar but different domain than the data it has initially been trained on. The goal of domain adaptation (DA) is to mitigate this domain shift problem by searching for an optimal feature transformation to learn a domain-invariant representation. Such a domain shift can appear in handwriting recognition (HWR) applications where the motion pattern of the hand and with that the motion pattern of the pen is different for writing on paper and on tablet. This becomes visible in the sensor data for online handwriting (OnHW) from pens with integrated inertial measurement units. This paper proposes a supervised DA approach to enhance learning for OnHW recognition between tablet and paper data. Our method exploits loss functions such as maximum mean discrepancy and correlation alignment to learn a domain-invariant feature representation (i.e., similar covariances between tablet and paper features). We use a triplet loss that takes negative samples of the auxiliary domain (i.e., paper samples) to increase the amount of samples of the tablet dataset. We conduct an evaluation on novel sequence-based OnHW datasets (i.e., words) and show an improvement on the paper domain with an early fusion strategy by using pairwise learning.

Keywords: Online handwriting recognition (OnHW) · sensor pen · domain adaptation (DA) · deep metric learning (DML) · writer-(in)dependent tasks.

1 Introduction

HWR can be categorized into offline and online HWR. While offline HWR deals with the analysis of the visual representation, OnHW recognition works on different types

Supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A and by the project “Schreibtrainer”, Grant No. 16SV8228, as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications within the framework of “BAYERN DIGITAL II”.

of spatio-temporal signals and can make use of temporal information such as writing direction and speed [20]. Typically, recording systems make use of a stylus pen together with a touch screen surface [1]. Systems for writing on paper became popular, first prototypical systems were used [4], and recently a novel system enhanced with inertial measurement units (IMUs) became prominent [19]. These IMU-enhanced pens are real-world applicable. While previous work [10, 15, 16, 18, 19] used this pen for writing on paper, [17] used this pen for writing on tablet. Figure 1 presents IMU data from a sensor-enhanced pen for writing on paper (left) and tablet (right). Due to the rough paper, the sensor data for writing on paper has more noise than writing on surface. Furthermore, the magnetic field of the tablet influences the magnetometer of the pen. This leads to different distributions of data and a domain shift between both data sources. Previously, tablet and paper data are processed separately, and hence, there is no method that can use both data sources simultaneously and inter-changeably.

Traditional ML algorithms assume training and test datasets to be *independent and identically distributed*. When applied in practice a domain shift appears in test data (here, shift between sensor data from tablet and paper), and hence, this assumption

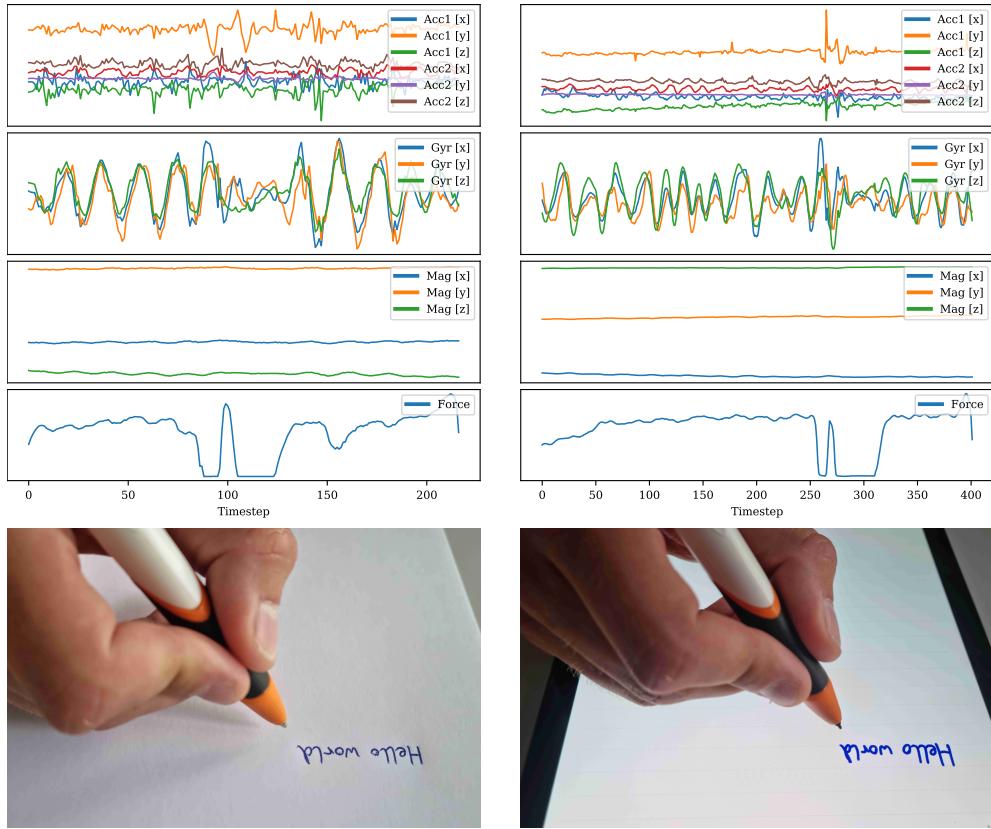


Fig. 1: Comparison of accelerometer (1st row), gyroscope (2nd row), magnetometer (3rd row) and force (4th row) data from a sensor pen [19] on paper (left) and tablet (right).

rarely holds in practice [23]. DA [12, 25] tries to compensate for this domain shift by transferring knowledge between both data sources. Most techniques transform the source data (here, data written on paper) by minimizing the distance to the target data (here, data written on tablet) [23], or by transforming the extracted features of the data sources [16]. To transform features of source domain into the target domain or to compare feature embeddings, higher-order moment matching (HoMM) [2] is often employed. Typically, the maximum mean discrepancy (MMD) [11] (HoMM of order 1) or the kernelized method kMMD [13] between features is evaluated. Correlation alignment (CORAL) [24] is of order 2. A related, yet different, task is pairwise learning. The *pairwise contrastive loss* [3] minimizes the distance between feature embedding pairs of the same class and maximizes the distance between feature embeddings of different classes dependent on a margin parameter. The *triplet loss* [21] defines an anchor and a positive as well as a negative point, and forces the positive pair distance to be smaller than the negative pair distance by a certain margin. While the triplet loss is typically used for image recognition, [9, 15] used this loss for sequence-based learning.

Contributions. We propose a method for OnHW recognition from sensor-enhanced pens for classifying words written on tablet and paper. We address the task of representation learning of features from different domains (i.e., tablet and paper) by using moment matching techniques (i.e., MMD and CORAL). For matching positive and negative samples of paper datasets w.r.t. anchor samples of tablet datasets, we use a triplet loss with dynamic margin and triplet selection based on the Edit distance. We conduct a large evaluation on OnHW [18] datasets. Website: www.iis.fraunhofer.de/de/ff/lv/data-analytics/anwproj/schreibtrainer/onhw-dataset.html.

The remainder of this paper is organized as follows. Section 2 discusses related work followed by our proposed methodology in Section 3. The experimental setup is described in Section 4 and the results are discussed in Section 5. Section 6 concludes.

2 Related Work

In this section, we address related work for OnHW recognition and for pairwise learning in relation to domain adaptation.

OnHW Recognition. The novel sensor-enhanced pen based on IMUs enables new applications for writing on normal paper. First, [19] introduced a character-based dataset from sensor-enhanced pens on paper and evaluated ML and DL techniques. [18] proposed several sequence-based datasets written on paper and tablet and a large benchmark of convolutional, recurrent and Transformer-based architectures, loss functions and augmentation techniques. To enhance the OnHW dataset with an offline HWR dataset, [15] generated handwritten images with ScrabbleGAN [7] and improved the training with cross-modal representation learning between online and offline HWR. [16] proposed a DA approach with optimal transport to adapt left-handed writers to right-handed writers for single characters. [17] reconstructed the trajectory of the pen tip for single characters written on tablet from IMU data and cameras pointing on the pen tip.

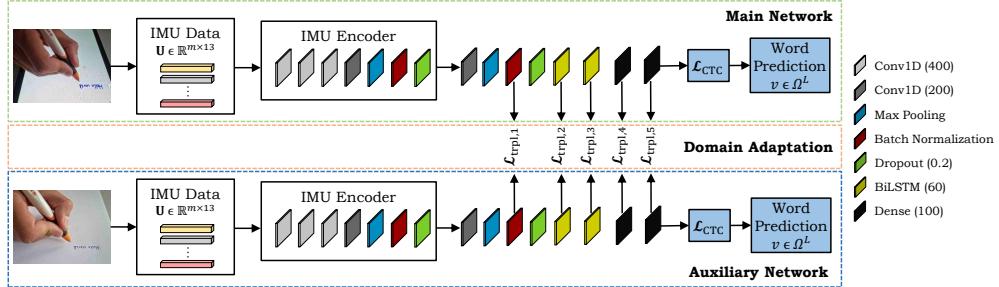


Fig. 2: **Detailed method overview:** The main network (top pipeline, tablet data) and the auxiliary network (bottom pipeline, paper data) consist of the respective pre-trained architectures with convolutional and bidirectional layers. The weights are fine-tuned with domain adaptation techniques such as the triplet loss at five different fusion points.

Pairwise Learning for DA. Research for pairwise and triplet learning is very advanced in general [3, 21], while the pairwise learning has rarely been used for sequence-based learning. [9] use a triplet selection with L_2 -normalization for language modeling. While they consider all negative pairs for triplet selection with fixed similarity intensity parameter, our triplet approach dynamically selects positive and negative samples based on ED that is closer to the temporally adaptive maximum margin function by [22] as data is evolving over time. DeepTripletNN [27] also uses the triplet loss on embeddings between time-series data (audio) and visual data. While their method uses cosine similarity for the final representation comparison, we make use of mean discrepancy and correlation techniques.

3 Methodology

We start with a formal definition of multivariate time-series (MTS) classification and an method overview in Section 3.1. We propose our sequence-based triplet loss in Section 3.2, and finally give details about DML for DA in Section 3.3.

3.1 Methodology

MTS Classification. We define the sensor data from pens with integrated IMUs as a MTS $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_m\} \in \mathbb{R}^{m \times l}$, an ordered sequence of $l \in \mathbb{N}$ streams with $\mathbf{u}_i = (u_{i,1}, \dots, u_{i,l}), i \in \{1, \dots, m\}$, where $m \in \mathbb{N}$ is the length of the time-series. The MTS training set is a subset of the array $\mathcal{U} = \{\mathbf{U}_1, \dots, \mathbf{U}_{n_U}\} \in \mathbb{R}^{n_U \times m \times l}$, where n_U is the number of time-series. Each MTS is associated with \mathbf{v} , a sequence of L class labels from a pre-defined label set Ω with K classes. For our classification task, $\mathbf{v} \in \Omega^L$ describes words. We train a convolutional neural network (CNN) in combination with a bidirectional long short-term memory (BiLSTM). We use the connectionist temporal classification (CTC) [8] loss to predict a word \mathbf{v} .

Method Overview. Figure 2 gives a method overview. The *main* task (top pipeline) is to classify sensor data represented as MTS with word labels \mathbf{v} written with a sensor-enhanced pen [19] on tablet. The *auxiliary* task (bottom pipeline) is to classify sensor data from the same sensor-enhanced pen written on paper. For optimally combining both datasets, we train a common representation between both networks by using the triplet loss $\mathcal{L}_{\text{trpl},c}$, see Section 3.2, with $c \in C = \{1, 2, 3, 4, 5\}$ defines the layer both networks are combined. $c = 1$ represents an intermediate fusion, while $c = 5$ represents a late fusion. With DML techniques, we minimize the distance (or maximizing the similarity) between the distributions of both domains (see Section 3.3).

3.2 Contrastive Learning and Triplet Loss

To learn a common representation typically pairs of same class labels of both domains are used. Pairs with similar but different labels can improve the training process. This can be achieved using the triplet loss [21] which enforces a margin between pairs of MTS of tablet and paper sources with the same identity to all other different identities. As a consequence, the feature embedding for one and the same labels lives on a manifold, while still enforcing the distance and thus discriminability to other identities. We define the MTS \mathbf{U}_i^a of the tablet dataset as *anchor*, an MTS \mathbf{U}_i^p of the paper dataset as the *positive* sample, and an MTS \mathbf{U}_i^n of the paper dataset as the *negative* sample. We seek to ensure that the embedding of the anchor $f_c(\mathbf{U}_i^a)$ of a specific label is closer to the embedding of the positive sample $f_c(\mathbf{U}_i^p)$ of the same label than it is to the embedding of any negative sample $f_c(\mathbf{U}_i^n)$ of another label. Thus, we want the inequality

$$\mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^p)) + \alpha < \mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^n)), \quad (1)$$

to hold for all training samples $(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^p), f_c(\mathbf{U}_i^n)) \in \Phi$ with Φ being the set of all possible triplets in the training set. α is a margin between positive and negative pairs. The DML loss \mathcal{L}_{DML} is defined in Section 3.3. The *contrastive loss* minimizes the distance of the anchor to the positive sample and separately maximizes the distance to the negative sample. Instead, we can formulate the *triplet loss* as

$$\begin{aligned} \mathcal{L}_{\text{trpl},c}(\mathbf{U}^a, \mathbf{U}^p, \mathbf{U}^n) = \sum_{i=1}^N \max & \left[\mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^p)) - \right. \\ & \left. \mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^n)) + \alpha, 0 \right], \end{aligned} \quad (2)$$

where N is the number of triplets. To ensure fast convergence, it is necessary to select triplets that violate the constraint from Equation 1. Computing the loss for all triplet pairs leads to poor training performance as poorly chosen pairs dominate hard ones [5]. We use the triplet selection approach by [15] that uses the Edit distance (ED) to define the identity and select triplets. We define two sequences with an ED of 0 as positive pair, and with an ED larger than 0 as negative pair (between 1 and 10). We use only substitutions for triplet selection. Figure 3 shows the number of triplet pairs for each ED. While there exist 265 samples for $ED = 0$, 3,022 samples for $ED = 1$ and 23,983 samples for $ED = 2$, the number of pairs highly increase for higher EDs.

For each batch, we search in a dictionary of negative sample pairs for samples with $ED = 1 + \lfloor \frac{\max_e - e - 1}{20} \rfloor$ as lower bound for the current epoch e and maximal epochs $\max_e = 200$ [15]. For every pair we randomly select one paper sample. We let the margin α in the triplet loss vary for each batch such that $\alpha = \beta \cdot \overline{ED}$ is depending on the mean ED of the batch and is in the range [1, 11]. β depends on the DML loss (see Section 3.3).

3.3 Domain Adaptation with Deep Metric Learning

A domain \mathcal{D} consists of a feature space \mathcal{X} with marginal probability $P(\mathcal{X})$. The task is defined by the label space \mathcal{Y} . When considering OnHW recognition, there is a source domain (paper dataset) $\mathcal{D}_S = \{\mathcal{U}_S^i, \mathcal{Y}_S^i\}_{i=1}^{\mathcal{N}_S}$ of \mathcal{N}_S labeled samples of $|\mathcal{Y}_S^i|$ categories, and a target domain (tablet dataset) $\mathcal{D}_T = \{\mathcal{U}_T^i, \mathcal{Y}_T^i\}_{i=1}^{\mathcal{N}_T}$ of \mathcal{N}_T labeled samples of $|\mathcal{Y}_T^i|$ categories. DA can mitigate the domain shift and improve the classification accuracy in the target domain by enforcing the distance of target embeddings $f_c(\mathbf{U}_i^a)$ and source domain embeddings $f_c(\mathbf{U}_i^p)$ and $f_c(\mathbf{U}_i^n)$

to be minimal. The embeddings are of size 400×200 for $c = 1$, of size 60×200 for $c = 2$ and $c = 3$, and of size 60×100 for $c = 4$ and $c = 5$. We search for a DML loss $\mathcal{L}_{\text{DML}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^p))$, respectively for the negative sample \mathbf{U}_i^n , that takes the domain shift into account. To perform domain alignment, we use higher-order moment matching (HoMM) [2]

$$\mathcal{L}_{\text{HoMM}}(f_c(\mathbf{U}_i^a), f_c(\mathbf{U}_i^p)) = \frac{1}{H^p} \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} f_c(\mathbf{U}_i^a)^{\otimes p} - \frac{1}{n_t} \sum_{i=1}^{n_t} f_c(\mathbf{U}_i^p)^{\otimes p} \right\|_F^2, \quad (3)$$

between embeddings $f_c(\mathbf{U}_i^a)$ and $f_c(\mathbf{U}_i^p)$, respectively for $f_c(\mathbf{U}_i^a)$ and $f_c(\mathbf{U}_i^n)$. It holds $n_s = n_t = b$ with the batch size b . $\|\cdot\|_F$ denotes the Frobenius norm, H is the number of hidden neurons in the adapted layer, and $(\cdot)^{\otimes p}$ denotes the p -level tensor power. When $p = 1$, HoMM is equivalent to the linear MMD [25], and when $p = 2$, HoMM is equivalent to the Gram matrix matching. When the embeddings are normalized by subtracting the mean, the centralized Gram matrix turns into the covariance matrix [2], and hence, HoMM for $p = 2$ is equivalent to CORAL [24]. However, the space complexity for calculating the tensor $(\cdot)^{\otimes p}$ reaches $\mathcal{O}(H^p)$. This can be reduced by *group moment matching* that divides the hidden neurons into n_g groups, with each group $\lfloor \frac{H}{n_g} \rfloor$ neurons, and the space complexity reduces to $\mathcal{O}(n_g \cdot \lfloor \frac{H}{n_g} \rfloor^p)$. Furthermore, *random sampling matching* randomly selects T values in the high-level tensor, and only aligns these T values in the source and target domains. The space complexity reduces to $\mathcal{O}(T)$ [2]. For our application, we evaluate orders $p = 1, p = 2$ and $p = 3$, and choose $T = 1,000$, which reaches the limits of our training setup of GPUs with 32 GB VRAM. Alternatively, we make use of (Jeff and Stein) CORAL [24]. We choose the hyperparameters β from Section 3.2 proposed in Table 1.

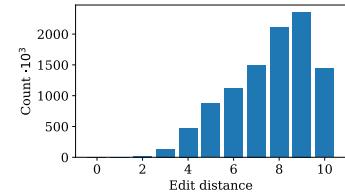


Fig. 3: Number tablet-paper pairs dependent on the ED.

Table 1: Hyperparameter choices of β for all DML loss functions and fusion points c .

DA Loss	$c = 1$	$c = 2$	$c = 3$	$c = 4$	$c = 5$
kMMD [13] ($p = 1$)	10	100	100	10	10
HoMM [2] ($p = 2$)	0.01	10^5	10^4	100	0.1
HoMM [2] ($p = 3$)	10^{-6}	10^6	10^5	100	10^{-3}
kHoMM [2] ($p = 2$)	10^3	10^6	10^6	10^4	10
kHoMM [2] ($p = 3$)	100	10^6	10^6	10^4	10
CORAL [23]	0.01	10^4	10^4	10	0.01
Jeff CORAL [23]	0.1	100	100	1	0.1
Stein CORAL [23]	1	100	100	10	1

4 Experiments

OnHW recognition uses time in association with different types of spatio-temporal signal. The pen in [19] uses two accelerometers (3 axes each), one gyroscope (3 axes), one magnetometer (3 axes), and one force sensor at 100 Hz. One sample of size $m \times l$ represents an MTS of m time steps from $l = 13$ sensor channels. We make use of three sequence-based datasets proposed by [18]: The *OnHW-words500* dataset contains 500 repeated words from 53 writers. The *OnHW-wordsRandom* contains randomly selected words from 54 writers. Both datasets combined represent the (auxiliary task) dataset from source domain written on paper, and contains in total 39,863 samples. The *OnHW-wordsTraj* dataset contains 4,262 samples of randomly selected words from two writers, and represents the (main task) dataset from target domain written on tablet. The challenging task is to adapt on one of the two writers (who collected data on tablet) by utilizing the paper datasets. We make use of 80/20 train/validation splits for writer-dependent (WD) and writer-independent (WI) evaluation.

Language Models (LMs). We apply LMs to the softmax output values of the neural networks. We use the Wikimedia database by the Wikimedia Foundation [26]. We create the n-gram dictionaries with the `nltk` package [14] and exclude punctuation marks. These dictionaries store the probabilities of the order of characters generated from sequences of items. Next, we select the paths (word length \times number of character labels) of the network predictions with the highest softmax values with a softmax threshold of 0.001. For more than $path_thresh = 512$ available paths, we limit the number of paths to $max_paths = 50$. Lastly, the n-gram models are applied to these paths.

5 Experimental Results

Hardware and Training Setup. For all experiments we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the vanilla Adam optimizer with a learning rate of 10^{-4} . We pre-train both networks for 1,000 epochs, and adapt for 200 epochs for the contrastive loss and 2,000 epochs for the triplet loss. A metric for sequence evaluation is the character error rate (CER) and the word error rate (WER) defined through the ED (see [18]).

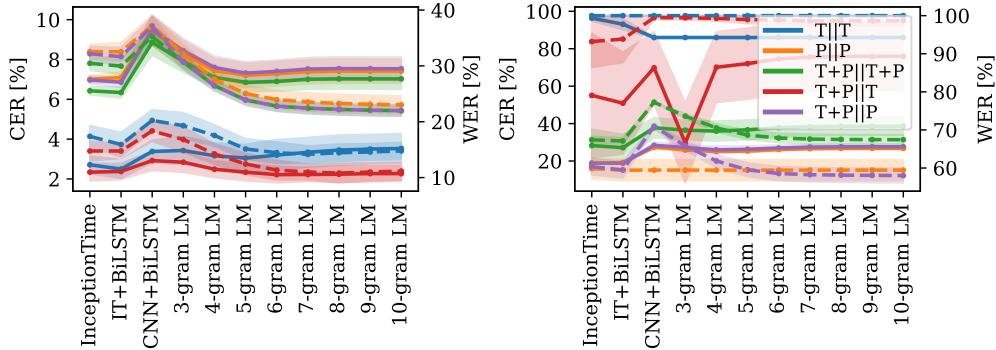


Fig. 4: Baseline results (WER: dashed, CER: solid, in %, averaged over cross-validation splits) for the InceptionTime (IT) [6] and our CNN+BiLSTM architectures and different n-gram LMs. Left: WD task. Right: WI task. Legend (tablet = T, paper = P): First notes training set and second notes validation set of the OnHW datasets [18].

5.1 Baseline Results

Figure 4 presents baseline results for our CNN+BiLSTM architecture compared to InceptionTime (IT) [6] with and without additional BiLSTM layer. Consistently, IT+BiLSTM outperforms IT, while the CER and WER slightly increases for our CNN+BiLSTM model. For all datasets, the WD classification task shows better performance than the WI task. We can improve the CNN+BiLSTM results with the LM from 3.38% to 3.04% CER and from 20.23% to 15.13% WER with 5-gram LM trained on the tablet dataset. While the WER consistently decreases with higher n-gram LM, the CER increases higher than 5-gram LM. This is more notable for the separate tablet (T) dataset as the length of the words are here shorter than for the paper (P) datasets. By simply combining both datasets, the models achieve lower error rates evaluated on the tablet dataset only (from 3.04% CER for T||T to 2.34% CER for T+P||T for 5-gram LM), but increases for the model evaluated on the paper dataset only (from 7.21% CER for P||P to 7.32% CER for T+P||P for 5-gram LM). We define X||Y, with X notes training dataset and Y notes validation dataset. This demonstrates the problem that there is a domain shift between tablet and paper data and that the size of the tablet dataset is small.

5.2 Domain Adaptation Results

We train the contrastive and pairwise learning approach by adapting paper data to tablet data with different representation loss functions (HoMM [2] and CORAL [24]) and propose results in Figure 5. State-of-the-art pairwise learning techniques cannot be applied to our setup as they are typically proposed for single label classification tasks. While the contrastive loss cannot improve the tablet validation results (5a), the validation on paper (5c) does improve (orange and purple curve of Figure 4). Also for the WI task, the paper validation improves (5d), while the tablet dataset is still a challenging task (5b). The triplet loss is on par with the baseline results for the WD task (5e). We see that early fusion ($c = 1$) leads to consistently low CERs as it is possible to adapt more

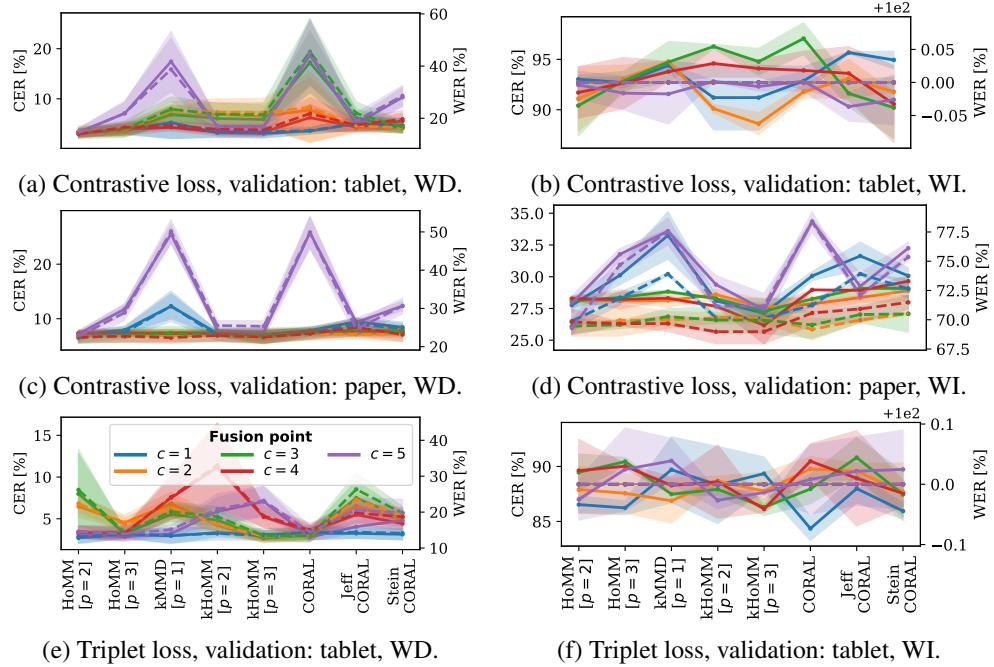


Fig. 5: DA results (WER: dashed, CER: solid, in %, averaged over cross-validation splits) for our CNN+BiLSTM architecture (with a 6-gram LM for tablet trainings, 10-gram for paper trainings respectively, for WD tasks and without LMs for WI tasks). The model is trained with the combined tablet and paper dataset with different representation loss functions at five different fusion points c .

trainable parameters after this fusion point. Intermediate ($c = 2$ and $c = 3$) and late ($c = 4$ and $c = 5$) fusion is dependent on the representation loss. kHoMM of order $p = 3$ at $c = 3$ leads to the highest significant improvement of 13.45% WER and 2.68% CER. The error rates of Jeff and Stein CORAL are marginally higher. LMs for $c = 4$ and $c = 5$ decrease results as the softmax output values are lower (uncertainty is higher) and the LM often chooses the second largest softmax value. We summarize the main difficulties as following: (1) While the paper dataset is very large, the tablet dataset as target domain is rather small. This leads to a small number of pairs with a small ED (see Figure 3). (2) Furthermore, as the OnHW-words500 dataset contains the same 500 word labels per writer, the variance of specific positive pairs is small.

6 Conclusion

We proposed a DA approach for online handwriting recognition for sequence-based classification tasks between writing on tablet and paper. For this, contrastive and triplet losses enhance the main dataset and allows a more generalized training process. We evaluated moment matching techniques as DML loss functions. The kernalized HoMM

of order 3 at the intermediate fusion layer combined with a n-gram language model provides the lowest error rates.

References

1. Alimoglu, F., Alpaydin, E.: Combining Multiple Representations and Classifiers for Pen-based Handwritten Digit Recognition. In: Intl. Conf. on Document Analysis and Recognition (ICDAR). vol. 2. Ulm, Germany (Aug 1997). <https://doi.org/10.1109/ICDAR.1997.620583>
2. Chen, C., Fu, Z., Chen, Z., Jin, S., Cheng, Z., Jin, X., sheng Hua, X.: HoMM: Higher-Order Moment Matching for Unsupervised Domain Adaptation. In: Proc. of the AAAI Conf. on Artificial Intelligence (AAAI). vol. 34(4), pp. 3422–3429 (Apr 2020)
3. Chopra, S., Hadsell, R., LeCun, Y.: Learning a Similarity Metric Discriminatively, with Application to Face Verification. In: Intl. Conf. on Computer Vision and Pattern Recognition (CVPR). San Diego, CA (Jun 2005). <https://doi.org/10.1109/CVPR.2005.202>
4. Deselaers, T., Keysers, D., Hosang, J., Rowley, H.A.: GyroPen: Gyroscopes for Pen-Input with Mobile Phones. In: IEEE Trans. Hum.-Mach. Syst. vol. 45(2), pp. 263–271 (Jan 2015). <https://doi.org/10.1109/THMS.2014.2365723>
5. Do, T.T., Tran, T., Reid, I., Kumar, V., Hoang, T., Carneiro, G.: A Theoretically Sound Upper Bound on the Triplet Loss for Improving the Efficiency of Deep Distance Metric Learning. In: Intl. Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 10404–10413. Long Beach, CA (Jun 2019). <https://doi.org/10.1109/CVPR.2019.01065>
6. Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weberf, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F.: InceptionTime: Finding AlexNet for Time Series Classification. In: arXiv preprint arXiv:1909.04939 (Sep 2019)
7. Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., Litman, R.: ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation. In: Intl. Conf. on Computer Vision and Pattern Recognition (CVPR). Seattle, WA (Jun 2020). <https://doi.org/10.1109/CVPR42600.2020.00438>
8. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A Novel Connectionist System for Unconstrained Handwriting Recognition. In: Trans. on Pattern Analysis and Machine Intelligence (TPAMI). vol. 31(5), pp. 855–868 (May 2009). <https://doi.org/10.1109/TPAMI.2008.137>
9. Guo, D., Tang, S., Wang, M.: Connectionist Temporal Modeling of Video and Language: A Joint Model for Translation and Sign Labeling. In: Intl. Joint Conf. on Artificial Intelligence (IJCAI). pp. 751–757 (2019). <https://doi.org/10.24963/ijcai.2019/106>
10. Klaß, A., Lorenz, S.M., Lauer-Schmaltz, M.W., Rügamer, D., Bischl, B., Mutschler, C., Ott, F.: Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift. In: IJCAI-ECAI Workshop on Spatio-Temporal Reasoning and Learning (STR) (Jul 2022)
11. Long, M., Cao, Y., Wang, L., Jordan, M.I.: Learning Transferable Features with Deep Adaptation Networks. In: Intl. Conf. on Machine Learning (ICML). vol. 37, pp. 97–105 (Jul 2015)
12. Long, M., Wang, J., Ding, G., Sun, J., Yu, P.S.: Transfer Joint Matching for Unsupervised Domain Adaptation. In: Intl. Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 1410–1417. Columbus, OH (Jun 2014). <https://doi.org/10.1109/CVPR.2014.183>
13. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep Transfer Learning with Joint Adaptation Networks. In: Intl. Conf. on Machine Learning (ICML). vol. 70, pp. 2208–2217 (Aug 2017)
14. NLTK: Natural Language Toolkit (Jul 2022), <https://www.nltk.org/index.html#>
15. Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C.: Cross-Modal Common Representation Learning with Triplet Loss Functions. In: arXiv preprint arXiv:2202.07901 (Feb 2022)

16. Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C.: Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In: Proc. of the ACM Intl. Conf. on Multimedia (ACMMM). pp. 5934–5943. Lisboa, Portugal (Oct 2022). <https://doi.org/10.1145/3503161.3548167>
17. Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C.: Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In: Proc. of the IEEE/CVF Winter Conf. for Applications on Computer Vision (WACV). pp. 266–276. Waikoloa, HI (Jan 2022). <https://doi.org/10.1109/WACV51458.2022.00131>
18. Ott, F., Rügamer, D., Heublein, L., Hamann, T., Barth, J., Bischl, B., Mutschler, C.: Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In: International Journal on Document Analysis and Recognition (IJ-DAR). vol. 25(12), p. 385–414 (Sep 2022). <https://doi.org/10.1007/s10032-022-00415-6>
19. Ott, F., Wehbi, M., Hamann, T., Barth, J., Eskofier, B., Mutschler, C.: The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In: Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT). vol. 4(3), article 92, pp. 1–20. Cancún, Mexico (Sep 2020). <https://doi.org/10.1145/3411842>
20. Plamondon, R., Srihari, S.N.: On-line and Off-line Handwriting Recognition: A Comprehensive Survey. In: Trans. on Pattern Analysis and Machine Intelligence (TPAMI). vol. 22(1), pp. 63–84 (Jan 2000). <https://doi.org/10.1109/34.824821>
21. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A Unified Embedding for Face Recognition and Clustering. In: Intl. Conf. on Computer Vision and Pattern Recognition (CVPR). Boston, MA (Jun 2015). <https://doi.org/10.1109/CVPR.2015.7298682>
22. Semedo, D., Magalhães, J.: Adaptive Temporal Triplet-loss for Cross-modal Embedding Learning. In: ACM Intl. Conf. on Multimedia (ACMMM). pp. 1152–1161 (Oct 2020). <https://doi.org/https://doi.org/10.1145/3394171.3413540>
23. Sun, B., Feng, J., Saenko, K.: Correlation Alignment for Unsupervised Domain Adaptation. In: arXiv preprint arXiv:1612.01939 (Dec 2016)
24. Sun, B., Saenko, K.: Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In: Europ. Conf. on Computer Vision (ECCV). vol. 9915, pp. 443–450 (Nov 2016). https://doi.org/https://doi.org/10.1007/978-3-319-49409-8_35
25. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep Domain Confusion: Maximizing for Domain Invariance. In: arXiv preprint arXiv:1412.3474 (Dec 2014)
26. Wikimedia Foundation: Wikimedia Downloads (Jul 2022), <https://dumps.wikimedia.org/>
27. Zeng, D., Yu, Y., Oyama, K.: Deep Triplet Neural Networks with Cluster-CCA for Audio-Visual Cross-Modal Retrieval. In: Trans. on Multimedia Computing, Communications, and Applications (TOMM). vol. 16(3), pp. 1–23 (Aug 2020). <https://doi.org/https://doi.org/10.1145/3387164>

Part III

Visual Self-Localization

Chapter 10

ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization

Contributing Article

Felix Ott, Tobias Feigl, Christoffer Löffler, and Christopher Mutschler. ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Joint Workshop on Long-Term Visual Localization, Visual Odometry and Geometric and Learning-based SLAM*, pages 187–198, Seattle, WA, June 2020. doi:10.1109/CVPRW50498.2020.00029.

Publisher Website

<https://ieeexplore.ieee.org/document/9150733>

Copyright License

© 2020 IEEE. Reprinted, with permission, from Felix Ott, Tobias Feigl, Christoffer Löffler, and Christopher Mutschler. ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 187–198, Seattle, WA, June 2020. doi:10.1109/CVPRW50498.2020.00029.

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of Ludwig-Maximilians University Munich products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how

to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing, and visualization. Tobias Feigl contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Christoffer Löffler contributed by the formal analysis (i.e., review and editing), visualization (i.e., presentation of the published work), data curation (i.e., data recording and data management), and supervision. Christopher Mutschler contributed with the computing resources, writing (i.e., review and editing), supervision, and funding acquisition by the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”. This paper was presented by Felix Ott at the IEEE/CVF International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) at the Joint Workshop on Long-Term Visual Localization, Visual Odometry and Geometric and Learning-based SLAM in June 2020¹⁸.

Statement on Differences between Master’s Thesis and Paper

The contributing paper is founded upon the master’s thesis titled “Komplementieren Relativer und Absoluter Eigenlokalisierungsverfahren” submitted on February 1, 2019, at the Friedrich-Alexander University Erlangen-Nürnberg by Ott (2019). Tobias Feigl and Michael Philippse served as supervisors for the aforementioned master’s thesis. The objective of the thesis was to integrate relative self-localization through optical flow with absolute self-localization based on images. As visual self-localization utilizing deep learning models was an emerging area of research in 2018, the master’s thesis provides a comprehensive summary of absolute self-localization techniques (i.e., simultaneous localization and mapping, visual odometry, structure from motion, and regression forests) while highlighting their respective advantages and drawbacks. Given the limited availability of publicly available datasets for visual localization, the master’s thesis offers a comprehensive review of all available datasets. Additionally, a section of thesis focuses on the acquisition of a significant dataset that was recorded in the Fraunhofer IIS L.I.N.K. test and application center to facilitate real-world evaluations. Moreover, synthetic datasets were created from a 3D model using Unity. The synthetic dataset mentioned herein consists of images, ground truth trajectories, optical flow, depth, and segmentations. Additionally, the thesis conducted a quantitative evaluation of various optical flow techniques concerning the application of relative pose estimation utilizing optical flow data from corresponding images. A comparison

¹⁸Video Link: <https://www.youtube.com/watch?v=ZCTxPJPCfF0>

was drawn between the fusion model utilizing both absolute and relative techniques and state-of-the-art fusion techniques employing Kalman filters. Furthermore, this paper introduces a novel visual odometry-aided pose estimation technique founded on the master’s thesis. The approach described herein is comprised of a time-distributed absolute pose regression model utilizing consecutive images, a relative pose regression model that estimates relative poses (position and orientation changes) through the prediction of optical flow via FlowNet2 (Ilg et al., 2017), and a fusion of both models with recurrent units to enhance the prediction of absolute pose. In the process of evaluating this approach, the Microsoft 7-Scenes (Shotton et al., 2013) dataset, the Warehouse (Löffler et al., 2018) dataset, and two datasets obtained during the course of the master’s thesis were utilized. The evaluation performed in the master’s thesis was limited to only four scenes (namely, chess, heads, office, and stairs) from the Microsoft 7-Scenes dataset. However, for the CVPR workshop paper, all scenes from the Microsoft 7-Scenes dataset were retrained. While the results for the Industry scenario #1 dataset in the contributing paper were derived from the master’s thesis, the results for the Industry scenario #2 and scenario #3 were retrained utilizing better model parameters. The CVPR workshop paper also featured an assessment of dynamic motions. Additionally, the paper included discussion on 6DoF pose regression with LSTMs, the influence of relative pose regression on the model, and runtimes considerations. The contributing paper was written anew, and the figures were recreated. Moreover, an extensive literature review was conducted independent of the master’s thesis.

Datasets

This paper uses the publicly available Industry scenario #1 dataset. We publish the Industry scenario #2 and Industry scenario #3 datasets that are available at:

<https://www.iis.fraunhofer.de/de/ff/lv/lok/opt1/warehouse.html>

https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets

Statement about Recent, Related Research¹⁷

As many image regions such as the sky, occlusions, and repetitive non-distinguishable pattern cannot be utilized for localization, these regions are of no use for the task, and hence, Altillawi (2022) avoids such image regions. This approach could be advantageous for the challenging Industry datasets (Ott et al., 2020a), since the L.I.N.K. hall environment contains many repetitive and texture-less features. In their work, Liao et al. (2021) proposed FINet, which builds a feature hierarchy by combining the shallow and deep layers of CNNs to capture more local appearance features. FINet has shown promising results in pose regression, outperforming the previously used absolute pose regression (APR) method (i.e., PoseNet). Another approach, Direct-PoseNet (Chen et al., 2021b), combines an APR network with view synthesis based direct matching. DFNet (Chen et al., 2022a) is a related method that combines APR with direct feature matching by addressing photometric distortions.

Musallam et al. (2022) introduce a translation and rotation equivariant CNN that can directly generate representations of camera motions in the feature space. The position and orientation Transformers (Shavit et al., 2021) consist of encoders, which aggregate activation maps using self-attention, and decoders, which transform latent features and scenes encodings into candidate pose predictions. To date, the method proposed by Shavit et al. (2021) is the only approach that uses a Transformer-based model for pose regression. Previous techniques learn the camera pose offline, whereas Cabrera-Ponce et al. (2021) propose a continual learning method for online relocalization to incorporate new acquired images associated with GPS coordinates. The use of additional sensors, such as LiDAR (light detection and ranging) sensors, can further enhance pose regression results, as demonstrated by STCLoc (Yu et al., 2022). However, this also incurs an additional cost of more expensive equipment.

ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization

Felix Ott¹, Tobias Feigl^{1,2}, Christoffer Löffler^{1,2}, and Christopher Mutschler^{1,3}

¹Fraunhofer Institute for Integrated Circuits IIS, Nuremberg, Germany

²Department of Computer Science, FAU Erlangen-Nuremberg, Germany

³Department of Statistics, Ludwig-Maximilians-University (LMU), Munich, Germany

{felix.ott | tobias.feigl | christoffer.loeffler | christopher.mutschler}@iis.fraunhofer.de

Abstract

Visual Odometry (VO) accumulates a positional drift in long-term robot navigation tasks. Although Convolutional Neural Networks (CNNs) improve VO in various aspects, VO still suffers from moving obstacles, discontinuous observation of features, and poor textures or visual information. While recent approaches estimate a 6DoF pose either directly from (a series of) images or by merging depth maps with optical flow (OF), research that combines absolute pose regression with OF is limited.

We propose ViPR, a novel modular architecture for long-term 6DoF VO that leverages temporal information and synergies between absolute pose estimates (from PoseNet-like modules) and relative pose estimates (from FlowNet-based modules) by combining both through recurrent layers. Experiments on known datasets and on our own Industry dataset show that our modular design outperforms state-of-the-art in long-term navigation tasks.

1. Introduction

Real-time tracking of mobile objects (e.g., forklifts in industrial areas) allows to monitor and optimize workflows and tracks goods for automated inventory management. Such environments typically include large warehouses or factory buildings, and localization solutions often use a combination of radio-, LiDAR- or radar-based systems, etc.

However, these solutions often require infrastructure or they are costly in their operation. An alternative approach is a (mobile) optical pose estimation based on ego-motion. Such approaches are usually based on SLAM (Simultaneous Localization and Mapping), meet the requirements of exact real-time localization, and are also cost-efficient.

Available pose estimation approaches are categorized into three groups: classical, hybrid, and deep learning (DL)-

based methods. Classical methods often require an infrastructure that includes either synthetic (i.e., installed in the environment) or natural (e.g., walls and edges) markers. The accuracy of the pose estimation depends to a large extent on suitable invariance properties of the available features such that they can be reliably recognized. However, to reliably detect features, we have to invest a lot of expensive computing time [38, 27]. Additional sensors (e.g., inertial sensors, depth cameras, etc.) or additional context (e.g., 3D models of the environment, prerecorded landmark databases, etc.) may increase the accuracy but also increase system complexity and costs [44]. Hybrid methods [66, 7, 6, 23, 74] combine geometric and machine learning (ML) approaches. For instance, ML predicts the 3D position of each pixel in world coordinates, from which geometry-based methods infer the camera pose [16].

Recent DL approaches partly address the above mentioned issues of complexity and cost, and also aim for high positioning accuracy, e.g., regression forests [51, 74] learn a mapping of images to positions based on 3D models of the environment. Absolute pose regression (APR) uses DL [63] as a cascade of convolution operators to learn poses only from 2D images. The pioneer PoseNet [33] has been extended by Bayesian approaches [31], long short-term memories (LSTMs) [77] and others [50, 26, 36, 11]. Recent APR methods such as VLocNet [72, 59] and DGRNets [42] introduce relative pose regression (RPR) to address the APR-

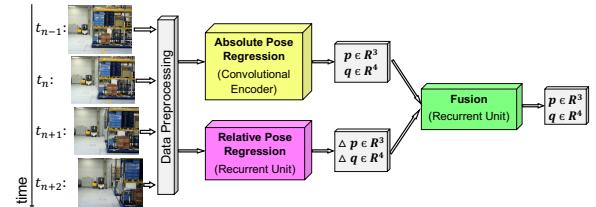


Figure 1: Our pose estimation pipeline solves the APR- and RPR-tasks in parallel, and recurrent layers estimate the final 6DoF pose.

task. While APR needs to be trained for a particular scene, RPR may be trained for multiple scenes [63]. However, RPR alone does not solve the navigation task.

For applications such as indoor positioning, existing approaches are not yet mature, i.e., in terms of robustness and accuracy to handle real-world challenges such as changing environment geometries, lighting conditions, and camera (motion) artifacts. This paper proposes a modular fusion technique for 6DoF pose estimation based on a PoseNet-like module and predictions of a relative module for VO. Our novel relative module uses the flow of image pixels between successive images computed by FlowNet2.0 [25] to capture time dependencies in the camera movement in the recurrent layers, see Fig. 1. Our model reduces the positioning error using this multitasking approach, which learns both the absolute poses based on monocular (2D) imaging and the relative motion for the task of estimating VO.

We evaluate our approach first on the small-scale 7-Scenes [66] dataset. As other datasets are unsuitable to evaluate continuous navigation tasks we also release a dataset that can be used to evaluate various problems arising from real industrial scenarios such as inconsistent lighting, occlusion, dynamic environments, etc. We benchmark our approach on both datasets against existing approaches [33, 77] and show that we consistently outperform the accuracy of their pose estimates.

The rest of the paper is structured as follows. Section 2 discusses related work. Section 3 provides details about our architecture. We discuss available datasets and introduce our novel *Industry* dataset in Section 4. We present experimental results in Section 5 before Section 6 concludes.

2. Related Work

SLAM-driven 3D point registration methods enable precise self-localization even in unknown environments. Although VO has made remarkable progress over the last decade, it still suffers greatly from scaling errors of real and estimated maps [43, 69, 49, 29, 34, 35, 40, 54, 4, 39]. With more computing power, Visual Inertial SLAM combines VO with Inertial Measurement Unit (IMU) sensors to partly resolve the scale ambiguity, to provide motion cues without visual features [43, 70, 29], to process more features, and to make tracking more robust [69, 34]. Multiple works combine global localization in a scene with SLAM/(Inertial) VO [46, 17, 55, 64, 22, 52, 28]. However, recent SLAM methods do not yet meet industry-strength with respect to accuracy and reliability [57, 18] as they need undamaged, clean and undisguised markers [39, 30] and as they still suffer from long-term stability and the effects of movement, sudden acceleration and occlusion [75]. SIFT-like point-based features [45] for the localization from landmarks [3, 24, 41, 78] require efficient retrieval methods, use VLAD encodings such as DenseVLAD [71], use anchor

points such as AnchorNet [60], or use RANSAC-based optimization such as DSAC [6] and ActiveSearch [61].

VO primarily addresses the problem of separating ego from feature-motion and suffers from area constraints, poorly textured environments, scale drift, a lack of an initial position, and thus inconsistent camera trajectories [10]. Instead, PoseNet-like architectures (see Sec. 2.1) that estimate absolute poses on single-shot images are more robust, less compute-intensive, and can be trained in advance on application data. Unlike VO, they do not suffer from a lack of initial poses and do not require access to camera parameters, good initialization, and handcrafted features [65]. Although the joint estimation of relative poses may contribute to increasing accuracy (see Sec. 2.2), such hybrid approaches still suffer from dynamic environments, as they are often trained offline in quasi-rigid environments. While optical flow (see Sec. 2.3) addresses these challenges it has not yet been combined with APR for 6DoF self-localization.

2.1. Absolute Pose Regression (APR)

Methods that derive a 6DoF pose directly from images have been studied for decades. Therefore, there are currently many classic methods whose complex components are replaced by machine learning (ML) or DL. For instance, RelocNet [2] learns metrics continuously from global image features through a camera frustum overlap loss. CamNet [15] is a coarse (image-based)-to-fine (pose-based) retrieval-based model that includes relative pose regression to get close to the best database entry that contains extracted features of images. NNet [37] queries a database for similar images to predict the relative pose between images and a RANSAC [67] solves the triangulation to provide a position. While those *classic* approaches have already been extended with DL-components their pipelines are expensive as they embed feature matching and projection and/or manage a database. Most recent (and simple) DL-based also outperform their accuracies.

The key idea of PoseNet [33] and its variants [32, 31, 20, 77, 76, 79, 58, 65, 56, 66] among others such as BranchNet [56] and Hourglass [66] is to use a CNN for camera (re-)localization. PoseNet works with scene elements of different scales and is partially insensitive to light changes, occlusions and motion blur. However, while Dense PoseNet [33] crops subimages, PoseNet2 [32] jointly learns network and loss function parameters, [31] links a *Bernoulli* function and applies variational inference [20] to improve the positioning accuracy. However, those variants work with single images, and hence, do not use the temporal context (which is available in continuous navigation tasks), that could help to increase accuracy.

In addition to PoseNet+LSTM [77], there are also similar approaches that exploit time-context that is inherently given by consecutive images (i.e., DeepVO [79],

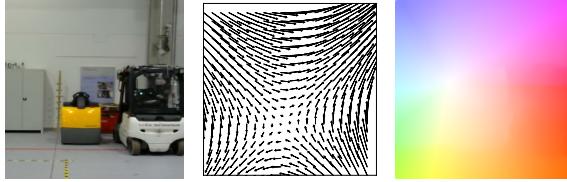


Figure 2: Optical flow (OF): input image (left); OF-vectors as RPR-input (middle); color-coded visualization of OF [1] (right).

ContextualNet [58], and VidLoc [12]). Here, the key-idea is to identify temporal connections in-between the feature vectors (extracted from images) with LSTM-units and to only track feature correlations that contribute the most to the pose estimation. However, there are hardly any long-term dependencies between successive images, and therefore LSTMs give worse or equal accuracy to, for example, simple averaging over successively estimated poses [65]. Instead, we combine estimated poses from time-distributed CNNs with estimates of the OF to maintain the required temporal context in the features of image series.

2.2. APR/RPR-Hybrids

In addition to approaches that derive a 6DoF pose directly from an image there are hybrid methods that combine them with VO to increase the accuracy. VLocNet [72] is closely related to our approach as it estimates a global pose and combines it with VO (but it does not use OF). To further improve the (re-)localization accuracy VLocNet++ [59] uses features from a semantic segmentation. However, we use different networks and do not need to share weights between VO and the global pose estimation. DGRNets [42] estimates both the absolute and relative poses, concatenates them, and uses recurrent CNNs to extract temporal relations between consecutive images. This is similar to our approach but we estimate the relative motion with OF, which allows us to train in advance on large datasets, making the model more robust. MapNet [8] learns a map representation from input data, combines it with GPS, inertial data, and unlabeled images, and uses pose graph optimization (PGO) to combine absolute and relative pose predictions. However, compared to all other methods the most accurate extension of it, MapNet+PGO, does not work on purely visual information, but exploits additional sensors.

2.3. Optical Flow

Typically, VO uses OF to extract features from image sequences. Motion fields, see Fig. 2 (middle), are used to estimate trajectories of pixels in a series of images. For instance, Flowdometry [53] and LS-VO [13] estimate displacements and rotations from OF. [48] proposed a VO-based dead reckoning system that uses OF to match features. [80] combined two CNNs to estimate the VO-motion: FlowNet2-ss [25] estimates the OF and PCNN [14]

links two images to process global and local pose information. However, to the best of our knowledge, we are the first to propose an OF-based architecture that estimates the relative camera movement through RNNs, using OF [25].

3. Proposed Model

After a data preprocessing that crops subimages of size $224 \times 224 \times 3$ from a sequence of four images, our pose regression pipeline consists of three parts (see Fig. 3): an APR-network, a RPR-network, and a 6DoF pose estimation (PE) network. PE uses the outputs of the APR- and RPR-networks to provide the final 6DoF pose.

3.1. Absolute Pose Regression (APR) Network

Our APR-network predicts the 6DoF camera pose from three input images based on the original PoseNet [33] model (i.e., essentially a modified GoogLeNet [68] with a regression head instead of a softmax) to train and predict the absolute positions $\mathbf{p} \in \mathbb{R}^3$ in the Euclidean space and the absolute orientations $\mathbf{q} \in \mathbb{R}^4$ as quaternions. From a single monocular image I the model predicts the pose

$$\hat{\mathbf{x}} = [\hat{\mathbf{p}}, \hat{\mathbf{q}}], \quad (1)$$

as approximations to the actual \mathbf{p} and \mathbf{q} . As the original model learns the image context, based on shape and appearance of the environment, but does not exploit the time context and relation between consecutive images [32], we adapted the model to a *time-distributed* variant. Hence, instead of a single image the new model receives three (consecutive) input images (at timesteps t_{n-1} , t_n , and t_{n+1}), see top part of Fig. 3, uses three separate dense layers (one for each pose) with 2,048 neurons each, and each of the dense layers yields a pose. The middle pose yields the most accurate position for the image at time step t_n .

3.2. Relative Pose Regression (RPR) Network

Our RPR-network uses FlowNet2.0 [25] on each consecutive pairs of the four input images to compute an approximation of the OF (see Fig. 2) and to predict three relative poses for later use. As displacements of similar length but from different camera viewing directions result in different OFs, the displacement and rotation of the camera between pairwise images must be *relative* to the camera's viewing direction of the first image. Therefore, we transform each camera's global coordinate systems (x_n, y_n, z_n) to the same local coordinate system $(\tilde{x}_n, \tilde{y}_n, \tilde{z}_n)$ by

$$\begin{pmatrix} \tilde{x}_n \\ \tilde{y}_n \\ \tilde{z}_n \end{pmatrix} = \mathbf{R} \begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix}, \quad (2)$$

with the rotation matrix \mathbf{R} . The displacement $\Delta\tilde{x}_n, \Delta\tilde{y}_n, \Delta\tilde{z}_n$ is the difference between the transformed coordinate systems. The displacement in global coordinates

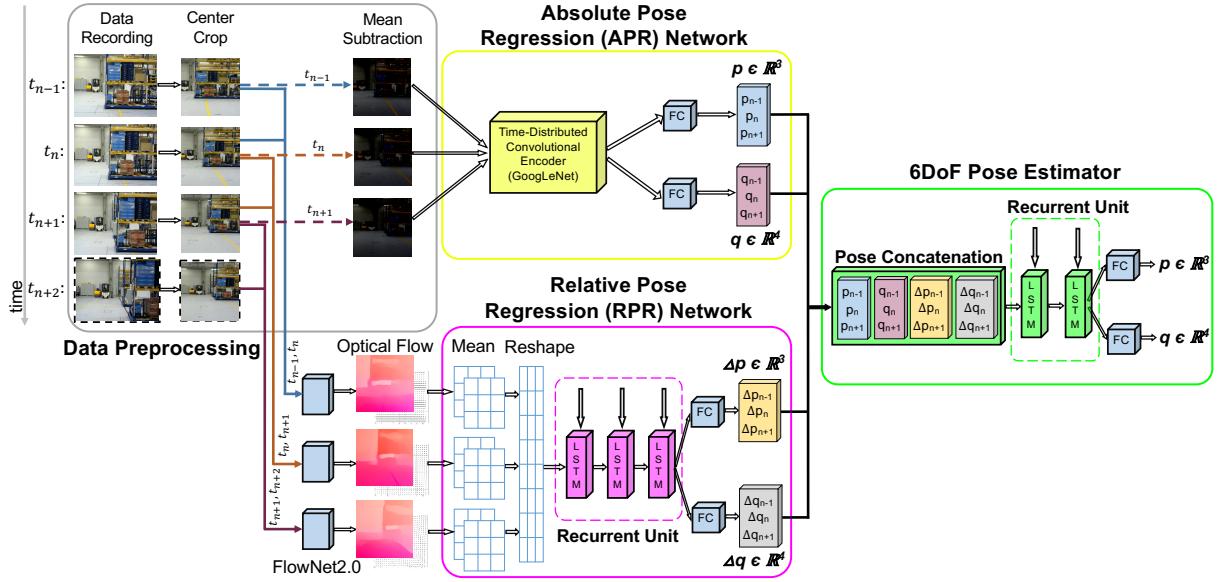


Figure 3: **Pipeline of the ViPR-architecture.** Data preprocessing (grey): Four consecutive input images (t_{n-1}, \dots, t_{n+2}) are center cropped. For the *absolute* network the mean is subtracted. For the *relative* network the OF is precomputed by FlowNet2.0 [25]. The *absolute* poses are predicted by our *time-distributed* APR-network (yellow). The RPR-network (purple) predicts the transformed *relative* displacements and rotations on reshaped mean vectors of the OF with (stacked) LSTM-RNNs. The PE modules (green) concatenate the absolute and relative modules and predicts the absolute 6DoF poses with stacked LSTM-RNNs.

is obtained by a back-transformation of the predicted displacement, such that

$$\mathbf{R}^T = \mathbf{R}^{-1} \text{ and } \mathbf{R}^T \mathbf{R} = \mathbf{R} \mathbf{R}^T = \mathbf{I}. \quad (3)$$

Fig. 4 shows the structure of the RPR-network. Similar to the APR-network, the RPR-network also uses a stack of images, i.e., three OF-fields from the four input images of the timesteps t_{n-1}, \dots, t_{n+2} , to include more time context.

In a preliminary study, we found that our recurrent units struggle to remember temporal features when the direct input of the OF is too large (raw size $224 \times 224 \times 3 \text{ px}$). This is in line with findings from Walch et al. [77]. Hence, we split the OF in zones and compute the mean value for each the u - and v -direction. We reshape 16×16 number of zones in both directions to the size 2×256 . The final concatenation results in a smaller total size of 3×512 . The LSTM-output is forwarded to 2 FC-layers that regress both the displacement (size 3×3) and rotation (size 3×4).

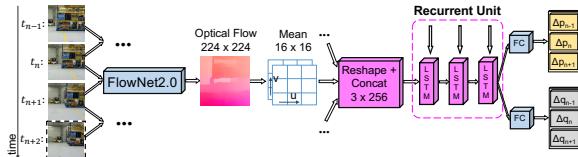


Figure 4: Pipeline of the *relative* pose regression (RPR) architecture: Data preprocessing, OF- and mean computation, reshaping, and concatenation, 3 recurrent LSTM units, and 2 FC-layers that yield the relative pose.

The 2 FC-layers use the following loss function to predict the relative transposed poses $\Delta\tilde{\mathbf{p}}^{tr}$ and $\Delta\tilde{\mathbf{q}}^{tr}$:

$$\mathcal{L} = \alpha_2 \|\Delta\tilde{\mathbf{p}}^{tr} - \Delta\mathbf{p}^{tr}\|_2 + \beta_2 \left\| \Delta\tilde{\mathbf{q}}^{tr} - \frac{\Delta\mathbf{q}^{tr}}{\|\Delta\mathbf{q}^{tr}\|_2} \right\|_2. \quad (4)$$

The first term accounts for the predicted and transformed displacement $\Delta\tilde{\mathbf{p}}^{tr}$ to the ground truth displacement $\Delta\mathbf{p}^{tr}$ with an L^2 -norm. The second term quantifies the error of the predicted rotation to the normalized ground truth rotation using an L^2 -norm. Both terms are weighted by the hyperparameters α_2 and β_2 . A preliminary grid search with a fixed $\alpha_2 = 1$ revealed an optimal value for β_2 that depends on the scaling of the environment.

3.3. 6DoF Pose Estimation (PE) Network

Our PE-network predicts absolute 6DoF poses from the outputs of both the APR- and RPR-networks, see Fig. 5. The PE-network takes as input the absolute position $\mathbf{p}_i = (x_i, y_i, z_i)$, the absolute orientation $\mathbf{q}_i = (w_i, p_i, q_i, r_i)$, the relative displacement $\Delta\mathbf{p}_i = (\Delta x_i, \Delta y_i, \Delta z_i)$, and the rotation change $\Delta\mathbf{q}_i = (\Delta w_i, \Delta p_i, \Delta q_i, \Delta r_i)$. As we feed poses from three sequential timesteps t_{n-1} , t_n , and t_{n+1} as input to the model it is implicitly *time-distributed*. The 2 stacked LSTM-layers and the 2 FC-layers return a 3DoF absolute position $\mathbf{p} \in \mathbb{R}^3$ and a 3DoF orientation $\mathbf{q} \in \mathbb{R}^4$ using the following loss:

$$\mathcal{L}(P, \Delta P) = \alpha_3 \|\tilde{\mathbf{p}} - \mathbf{p}\|_2 + \beta_3 \left\| \tilde{\mathbf{q}} - \frac{\mathbf{q}}{\|\mathbf{q}\|_2} \right\|_2. \quad (5)$$

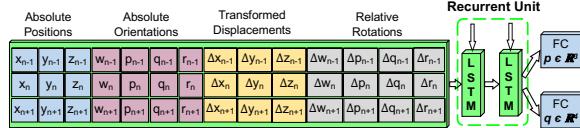


Figure 5: Pipeline of the 6DoF PE-architecture. The input tensor (3×14) contains absolute positions and orientations and relative displacements and rotations at timesteps t_{n-1}, t_n, t_{n+1} . 2 stacked LSTMs process the tensor and 2 FC-layers return the pose.

Again, in a preliminary grid search we chose L_2 -norms with a fixed $\beta_3 = 1$ that revealed an optimal value for α_3 .

4. Evaluation Datasets

To train our network we need two different types of image data: (1) images annotated with their absolute poses for the APR-network, and (2) images of OF, annotated with their relative poses for the RPR-network.

Datasets to evaluate APR. Publicly available datasets for absolute pose regression (Cambridge Landmarks [33] and TUM-LSI [77]) either lack accurate ground truth labels or the proximity between consecutive images is too large to embed meaningful temporal context. The Aalto University [37], Oxford RobotCar [47], DeepLoc [59] and CMU Seasons [62] datasets solve the small-scale issue of the 7-Scenes [66] dataset, but are barely used for evaluation of state-of-the-art techniques or consider only automotive-driving scenarios. The 12-Scenes [73] dataset is only used by DSAC++ [5]. For our industrial application these datasets are insufficient. 7-Scenes [66] only embeds scenes with less training data and only enables small scene-wise evaluations, but is mainly used for evaluation. Hence, to compare ViPR with recent techniques we use the 7-Scenes [66] dataset. Furthermore, we recorded the *Industry* dataset (see Sec. 4.1) that embeds three different industrial-like scenarios to allow a comprehensive and detailed evaluation with different movement patterns (such as slow motion and fast rotation).

Datasets to evaluate RPR. To evaluate the performance of the RPR and its contribution to ViPR, we also need a dataset with a close proximity between consecutive images. This is key to calculate the relative movement with OF. However, most publicly available datasets (Middlebury [1], MPI Sintel [9], KITTI Vision [21], and FlyingChairs [19]) either do not meet this requirement or the OF pixel velocities do not match those of real-world applications. Hence, we directly calculate the OF from images with FlowNet2.0 [25] to train the RPR on it. Our novel *Industry* dataset allows this, while retaining a large, diverse environment with hard real-world conditions, as described in the following.

4.1. Industry Dataset

We designed the *Industry* dataset to suite the requirements of both the APR- and the RPR-network and published the data¹ at large-scale ($1,320m^2$) using a high-precision ($< 1mm$) laser-based reference system. Each scenario presents different challenges (such as dynamic ego-motion with motion blur), various environmental characteristics (such as different geometric scales, light changes, i.e., artificial and natural light), and ambiguously structured elements, see Fig. 6.

Industry Scenario #1 [44] has been recorded with 8 cameras (approx. 60° field-of-view (FoV) each) mounted on a stable apparatus to cover 360° (with overlaps) that has been moved automatically at a constant velocity of approx. $0.3 m/s$. The height of the cameras is at $1.7 m$. The scenario contains 521,256 images ($640 \times 480 px$) and densely covers an area of $1,320 m^2$. The environment imitates a typical warehouse scenario under realistic conditions. Besides well-structured elements such as high-level racks with goods, there are also very ambiguous and homogeneously textured elements (e.g., blank white or dark black walls). Both natural and artificial light illuminates volatile structures such as mobile work benches. While the training dataset is composed of a horizontal and vertical zig-zag movement of the apparatus the test datasets movements vary to cover different properties for a detailed evaluation, e.g., different environmental scalings (i.e., *scale transition*, *cross*, *large scale*, and *small scale*), network generalization (i.e., *generalize open*, *generalize racks*, and *cross*), fast rotations (i.e., *motion artifacts* was recorded on a forklift at $2.26 m$ height) and volatile objects (i.e., *volatility*).

Industry Scenario #2 uses three 170° cameras (with overlaps) on the same apparatus at the same height. The recorded 11,859 training images ($1,280 \times 720 px$) represent a horizontal zig-zag movement (see Fig. 7a) and 3,096 test images represent a diagonal movement (see Fig. 7b). Compared to Scenario #1 this scenario has more variation in its velocities (between $0 m/s$ and $0.3 m/s$, SD $0.05 m/s$).

Industry Scenario #3 uses four 170° cameras (with overlaps) on a forklift truck at a height of $2.26 m$. Both the training and test datasets represents camera movements at varying, faster, and dynamic speeds (between $0 m/s$ and $1.5 m/s$, SD $0.51 m/s$). This makes the scenario the most challenging one. The training trajectory (see Fig. 7c) consists of 4,166 images and the test trajectory (see Fig. 7d) consists of 1,687 images. In contrast to the Scenarios #1 and #2 we train and test a typical industry scenario on dynamic movements of a forklift truck. However, one of cameras' images were corrupted in the test dataset, and thus, not used in the evaluation.

¹Industry dataset available at: <https://www.iis.fraunhofer.de/warehouse>. Provided are **raw images** and corresponding labels: p and q .

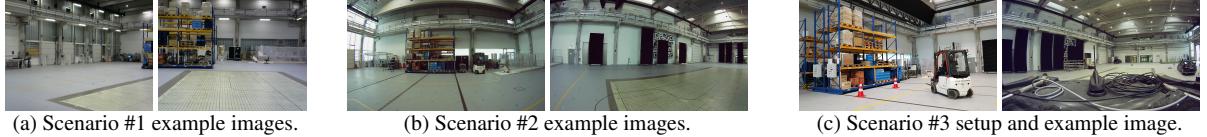


Figure 6: **Industry datasets.** Setup of the measurement environment (i.e., forklift truck, warehouse racks and black walls) and example images with normal (a) and wide-angle (b+c) cameras.

5. Experimental Results

To compare ViPR with state-of-the-art results, we first briefly describe our parameterization of PoseNet [33] and PoseNet+LSTM [77] in Sec. 5.1. Next, Sec. 5.2 presents our results. We highlight the performance of ViPR’s sub-networks (APR, APR+LSTM) individually, and investigate both the impact of RPR and PE on the final pose estimation accuracy of ViPR. Sec. 5.3 shows results of the RPR-network. Finally, we discuss general findings and show runtimes of our models in Sec. 5.4.

For all experiments we used an AMD Ryzen 7 2700 CPU 3.2 GHz equipped with one NVidia GeForce RTX 2070 with 8 GB GDDR6 VRAM. Tab. 1 shows the median error of the position in m and the orientation in $degrees$. The second column reports the spatial extends of the datasets. The last column reports the improvement in position accuracy of ViPR (in %) over APR-only.

5.1. Baselines

As a baseline we report the initially described results on 7-Scenes of PoseNet [33] and PoseNet+LSTM [77] (*in italic*). We further re-implemented the initial variant of PoseNet and trained it from scratch with $\alpha_1 = 1$, $\beta_1 = 30$ (thus optimizing for positional accuracy at the expense of orientation accuracy). Tab. 1 (cols. 3 and 4) shows our implementation’s results next to the initially reported ones (on 7-Scenes). We see that (as expected) the results of the PoseNet implementations differ due to changed values for α_1 and β_1 in our implementation.

5.2. Evaluation of the ViPR-Network

In the following, we evaluate our method in multiple scenarios with different distinct challenges for the pose estimation task. 7-Scenes focuses on difficult motion blur conditions of typical human motion. We then use the *Industry Scenario #1* to investigate various challenges at a larger scale, but with mostly constant velocities. *Industry Scenarios #2* and *#3* focus on dynamic, fast ego-motion at large-scale.

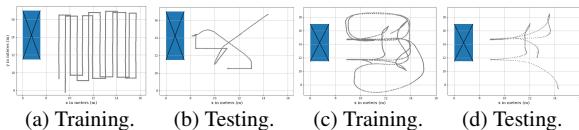


Figure 7: Exemplary trajectories of *Industry Scenarios #2* (a-b) and *#3* (c-d) to assess the generalizability of ViPR.

ios #2 and #3 then focus on dynamic, fast ego-motion of a moving forklift truck at large-scale.

7-Scenes [66]. For both architectures (PoseNet and ViPR), we optimized β to weight the impact of position and orientation such that it yields the smallest total median error. Both APR+LSTM and ViPR return a slightly lower pose estimation error of $0.33\ m$ and $0.32\ m$ than PoseNet+LSTM with $0.34\ m$. ViPR yields an average improvement of the position accuracy of 3.18% even in strong motion blur situations. The results indicate that ViPR relies on a plausible optical flow component to achieve performance that is superior to the baseline. In situations of negligible motion between frames the median only improves by $0.02\ m$. However, the average accuracy gain still shows that ViPR performs *en par* or better than the baselines.

Stable motion evaluation. For the *Industry Scenario #1* dataset, we train the models on the zig-zag trajectories, and test them on specific sub-trajectories with individual challenges, but at almost constant velocity. In total, ViPR improves the position accuracy by 12.27% on average (min.: 4.03% ; max.: 25.31%) while the orientation error is similar for most of the architectures and test sets.

In environments with volatile features, i.e., objects that are only present in the test dataset, we found that ViPR (with optical flow) is significantly (6.41%) better compared to APR-only. However, the high angular error of 77.54° indicates an irrecoverable degeneration of the APR-part. In tests with different scaling of the environment, we think that ViPR learns an interpretation of relative and absolute position regression, that works both in small and large proximity to environmental features, as ViPR improves by 15.52% (scale trans.) and 14.41% (small scale) or 10.68% (large scale). When the test trajectories are located within areas that embed only few or no training samples (gener. racks and open), ViPR still improves over other methods with $4.03\text{--}11.75\%$. The highly dynamic test on a forklift truck (motion artifacts) is exceptional here as only the test dataset contains dynamics and blur, and hence, challenges ViPR most. However, ViPR still improves by 10.01% over APR-only, despite the data dynamic’s absolute novelty.

In summary, ViPR decreases the position median significantly by about $2.53\ m$ than only APR+LSTM ($4.89\ m$). This and the other findings are strong indicators that the relative component RPR significantly supports the final pose estimation of ViPR.

Table 1: Pose estimation results (position and orientation median error in meters m and degrees ($^\circ$)) and total improvement of PE in % on the 7-Scenes [66] and *Industry* datasets. The best results are bold and underlined ones are additionally referenced in the text.

Dataset	Spatial extend (m)	PoseNet [33] (original/our param.)	PoseNet+LSTM [77]	APR-only	APR+LSTM (our param.)	ViPR*	Improv. ViPr (%)						
7-Scenes [66]	chess	$3.0 \times 2.0 \times 1.0$	$0.32/0.24$	$4.06/7.79$	0.24	5.77	0.27	9.66	0.22	7.89	+ 1.74		
	fire	$2.5 \times 1.0 \times 1.0$	$0.47/0.39$	$14.4/12.40$	<u>0.34</u>	<u>11.9</u>	0.39	12.85	0.50	15.70	0.38	12.74	+ 2.56
	heads	$2.0 \times 0.5 \times 1.0$	$0.29/0.21$	$6.00/16.46$	<u>0.21</u>	<u>13.7</u>	0.22	16.48	0.23	16.91	0.21	16.41	+ 3.64
	office	$2.5 \times 2.0 \times 1.5$	$0.48/0.33$	$3.84/10.08$	<u>0.30</u>	<u>8.08</u>	0.36	10.11	0.37	10.83	0.35	9.59	+ 4.01
	pumpkin	$2.5 \times 1.0 \times 1.0$	$0.47/0.45$	$8.42/8.70$	<u>0.33</u>	<u>7.00</u>	0.39	8.57	0.86	49.46	0.37	8.45	+ 5.12
	red kitchen	$4.0 \times 3.0 \times 1.5$	$0.59/0.41$	$8.64/9.08$	<u>0.37</u>	<u>8.83</u>	0.42	9.33	1.06	50.67	0.40	9.32	+ 4.76
	stairs	$2.5 \times 2.0 \times 1.5$	$0.47/0.36$	$6.93/13.69$	<u>0.40</u>	<u>13.7</u>	0.31	12.49	0.42	13.50	0.31	12.65	+ 0.46
	∅ total		<u>0.44/0.34</u>	<u>7.47/11.17</u>	<u>0.31</u>	<u>9.85</u>	<u>0.33</u>	11.11	0.53	23.82	0.32	11.01	+ 3.18
Industry Scenario 1 [44]	cross	24.5×16.0	$-/1.15$	$-/0.75$	—	0.61	0.53	4.42	0.21	0.46	0.60	+ 25.31	
	gener. open	20.0×17.0	$-/1.94$	$-/11.73$	—	1.68	11.07	3.36	2.95	1.48	10.86	+ 11.75	
	gener. racks	8.5×18.5	$-/3.48$	$-/6.01$	—	2.48	1.53	3.90	0.61	2.38	1.95	+ 4.03	
	large scale	19.0×19.0	$-/2.32$	$-/6.37$	—	2.37	9.82	4.99	1.61	2.12	8.64	+ 10.68	
	motion art.	37.0×17.0	$-/7.43$	$-/124.94$	—	7.48	131.30	8.18	139.37	6.73	136.6	+ 10.01	
	scale trans.	28.0×19.5	$-/2.17$	$-/3.03$	—	1.94	6.46	5.63	0.58	1.64	6.29	+ 15.52	
	small scale	10.0×11.0	$-/3.78$	$-/9.18$	—	4.09	20.75	4.46	6.06	3.50	15.74	+ 14.41	
	volatility	29.0×13.0	$-/2.68$	$-/78.52$	—	2.09	77.68	4.16	78.73	1.96	77.54	+ 6.41	
	∅ total		$-/3.12$	$-/30.07$	—	2.82	32.30	4.89	28.76	2.53	32.28	+ 12.27	
Industry Scen. 2	cam #0	6.5×9.0	$-/0.49$	$-/0.21$	—	0.22	0.29	1.49	0.14	0.16	3.37	+ 26.24	
	cam #1	6.5×9.0	$-/0.15$	$-/0.38$	—	0.23	0.35	2.68	0.17	0.12	2.75	+ 46.49	
	cam #2	6.5×9.0	$-/0.43$	$-/0.19$	—	0.37	0.13	0.90	0.15	0.30	1.84	+ 17.87	
	∅ total		$-/0.36$	$-/0.26$	—	<u>0.27</u>	<u>0.26</u>	<u>1.69</u>	<u>0.15</u>	0.20	2.65	+ 30.20	
Industry Scen. 3	cam #0	6.0×11.0	$-/0.41$	$-/1.00$	—	0.34	1.26	0.72	1.31	0.27	1.43	+ 20.64	
	cam #1	6.0×11.0	$-/0.32$	$-/1.07$	—	0.26	1.11	0.88	1.27	0.21	1.06	+ 20.13	
	cam #2	6.0×11.0	$-/0.32$	$-/1.60$	—	0.36	1.62	0.72	1.74	0.32	1.38	+ 11.47	
	∅ total		$-/0.35$	$-/1.22$	—	0.32	1.33	0.77	1.44	0.27	1.29	+ 17.41	

Industry Scenario #2 is designed to evaluate for unknown trajectories. Hence, training trajectories represent an orthogonal grid, and test trajectories are diagonal. In total, ViPR improves the position accuracy by 30.2 % on average (min.: 17.87 %; max.: 46.49 %). Surprisingly, the orientation error is comparable for all architectures, except ViPR. We think that this is because ViPR learns to optimize its position based on the APR- and RPR- orientations, and hence, exploits these orientations to improve its position estimate, that we prioritized in the loss function. APR-only yields an average position accuracy of 0.27 m, while the pure PoseNet yields position errors of 0.36 m on average, but APR+LSTM results in an even worse accuracy of 1.69 m. Instead, the novel ViPR outperforms all significantly with 0.2 m. Compared to our APR+LSTM approach, we think that ViPR on the one hand interprets and compen-

sates the (long-term) drift of RPR and on the other hand smooths the short-term errors of APR, as PE counteracts the accumulation of RPR’s scaling errors with APR’s absolute estimates. Here, the synergies of the networks in ViPR are particularly effective. This is also visualized in Fig. 8a: the green (ViPR) trajectory aligns more smoothly to the blue baseline when the movement direction changes. This also indicates that the RPR component is necessary to generalize to unknown trajectories and to compensate scaling errors.

Dynamic motion evaluation. In contrast to the other datasets, the *Industry Scenario #3* includes fast, large-scale, and high dynamic ego-motion in both training and test datasets. However, all estimators result in similar findings as *Scenario #2* as both scenarios embed motion dynamics and unknown trajectory shapes. Accordingly, ViPR again improves the position accuracy by 17.41 % on average (min.: 11.47 %; max.: 20.64 %), but this time exhibits very similar orientation errors. Improved orientation accuracy compared to *Scenario #2* is likely due to diverse orientations available in this dataset’s training.

Fig. 8b shows exemplary results that visualize how ViPR handles especially motion changes and motion dynamics (see the abrupt direction change between $x \in [8 - 9]$ m and $y \in [14 - 16]$ m). The results also indicate that ViPR predicts the smoothest and most accurate trajectories on unknown trajectory shapes (compare the trajectory segments between $x \in [11 - 12]$ m and $y \in [14 - 16]$ m). We think the reason why ViPR significantly outperforms APR

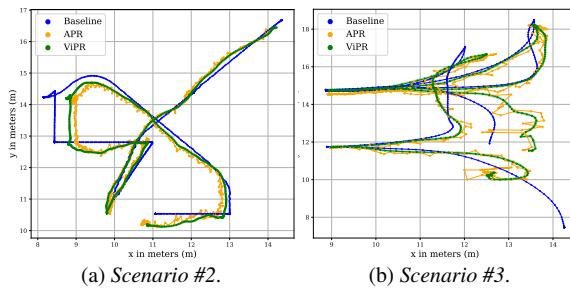


Figure 8: Exemplary comparison of APR, ViPR, and a baseline (ground truth) trajectory of the *Industry* datasets.

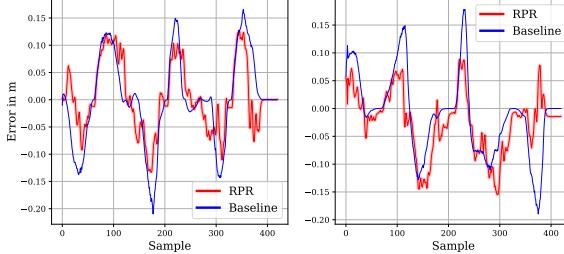


Figure 9: Exemplary **RPR-results** (displacements m) against the **baseline** (ground truth) on the *Scenario #3* dataset (see Fig. 7d).

by 20.13 % here is because of the synergy of APR, RPR, and PE. RPR contributes most in fast motion-changes, i.e., in motion blur situation. The success of RPR may also indicate that RPR differentiates between ego- and feature-motion to more robustly estimate a pose.

5.3. Evaluation of the RPR-Network

We use the smaller FlowNet2-s [25] variant of FlowNet2.0 as this has faster runtimes (140 Hz), and use it pretrained on the FlyingChairs [19], ChairsSDHom and FlyingThings3D datasets. To highlight that RPR contributes to the accuracy of the final pose estimates of ViPR, we explicitly test it on the *Industry Scenario #3* that embeds dynamic motion of both ego- and feature-motion. The distance between consecutive images is up to 20 cm, see Fig. 9. This results in a median error of 2.49 cm in x - and 4.09 cm in y -direction on average (i.e., the error is between 12.5 % and 20.5 %). This shows that the RPR yields meaningful results for relative position regression in a highly dynamic and difficult setting. It furthermore appears to be relatively robust in its predictions despite both ego- and feature-motion.

5.4. Discussion

6DoF Pose Regression with LSTMs. APR-only increases the positional accuracy over PoseNet for all datasets, see Tab. 1. We found that the position errors increase when we use methods with independent and single-layer LSTM-extensions [77, 79, 58, 65] on both the 7-Scenes and the *Industry* datasets, by 0.04 m up to 2.07 m. This motivated us to investigate stacked LSTM-layers only for the RPR- and PE-networks. We support the statement of Seifi et al. [65] that the motion between consecutive frames is too small, and thus, naive CNNs are already unable to embed them. Hence, additionally connected LSTMs are also unable to discover and track meaningful temporal and contextual relations between the features.

Influence of RPR to ViPR. To figure out the information gain of the RPR-network we also constructed ViPR in a closed end-to-end architecture through direct concatenation of the CNN-encoder-output (APR) and the LSTM-output (RPR). For a smaller OF-input (3×3) of the RPR-model the accuracy of the 7-Scenes [66] dataset increases, but

decreases for the *Industry* dataset. This stems from the fact that the relative movements of the 7-Scenes dataset are too small (< 2 cm) compared to the *Industry* dataset (approx. 20 cm). Hence, ViPR’s contribution is limited here.

Comparison of ViPR to state-of-the-art methods.

VLocNet++ [59] currently achieves the best results on 7-Scenes [66], but due to the small relative movement and the high ground truth error compared to VLocNet’s results a plausible evaluation is not possible regarding industrial applications. MapNet [8] achieves (on average) better results than ViPR on the 7-Scenes dataset, but results in a similar error, e.g., 0.30 m and 12.08° on the *stairs* set against ViPR’s 0.31 m and 12.65°. MapNet has an improvement of 8.7 % over PoseNet2 [32] and achieves 41.4 m and 12.5° on the RobotCar [47] dataset. However, a fair evaluation on this dataset with state-of-the-art methods requires results and code from VLocNet [72, 59].

Runtimes. The training of the APR takes 0.86 s per iteration for a batch size of 50 (GoogLeNet [68]) on our hardware setup. The training of the RPR and PE is faster (0.065 s) even at a higher batch size of 100, as these models are smaller (214,605, resp. 55,239, parameters). Hence, it is possible to retrain the PE-network quickly upon environment changes. The inference time of ViPR is between 7 ms and 9 ms per sample (PoseNet: avg. 5 ms, FlowNet2-s: avg. 9 ms). In addition, ViPR does not require domain knowledge to provide scenario-dependent applicability, nor does it need a compute-intensive matcher like brute force or RANSAC [67, 6]. However, instead of PoseNet, ViPR can also use such classical approaches in its modular process pipeline. DenseVLAD [71] and classical approaches are 10x (200-350 ms/sample) more computationally intensive than today’s deep pose regression variants.

6. Conclusion

In this paper, we addressed typical challenges of learning-based visual self-localization of a monocular camera. We introduced a novel DL-architecture that makes use of three modules: an absolute and a relative pose regressor module, and a final regressor that predicts a 6DoF pose by concatenating the predictions of the two former modularities. To show that our novel architecture improves the absolute pose estimates, we compared it with a publicly available dataset and proposed novel *Industry* datasets that enable a more detailed evaluation of different (dynamic) movement patterns, generalization, and scale transitions.

Acknowledgements

This work was supported by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGI-TAL II".

References

[1] Simon Baker, Stefan Roth, Daniel Scharstein, Michael J. Black, J. P. Lewis, and Richard Szeliski. "A Database and Evaluation Methodology for Optical Flow". In *Intl. Conf. on Computer Vision (ICCV)*, pages 1–8, Rio de Janeiro, Brazil, 2007. 3, 5

[2] Vassileios Balntas, Shuda Li, and Victor Prisacariu. "RelocNet: Continuous Metric Learning Relocalisation Using Neural Nets". In *Europ. Conf. on Computer Vision (ECCV)*, 2018. 2

[3] Alessandro Bergamo, Sudipta N. Sinha, and Lorenzo Torresani. "Leveraging Structure from Motion to Learn Discriminative Codebooks for Scalable Landmark Classification". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 763–770, Portland, OR, 2013. 2

[4] James Bergen. Visual odometry. *Intl. Journal of Robotics Research*, 33(7):8–18, 2004. 2

[5] Eric Brachman and nCarsten Rother. "Expert Sample Consensus Applied to Camera Re-Localization". In *Intl. Conf. on Computer Vision (ICCV)*, Oct. 2019. 5

[6] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel Stefan Gumhold, and Carsten Rother. "DSAC — Differentiable RANSAC for Camera Localization". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2492–2500, Honolulu, HI, 2017. 1, 2, 8

[7] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. "Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3364–3372, Las Vegas, NV, 2016. 1

[8] Samarth Brahmabhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. "Geometry-Aware Learning of Maps for Camera Localization". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2616–2625, Salt Lake City, UT, 2018. 3, 8

[9] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. "A Naturalistic Open Source Movie for Optical Flow Evaluation". In *Proc. Europ. Conf. on Computer Vision (ECCV)*, pages 611–625, Florence, Italy, 2012. 5

[10] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *Trans. on Robotics*, 32(6):1309–1332, 2016. 2

[11] Ming Chi, Chunhua Shen, and Ian Reid. "A Hybrid Probabilistic Model for Camera Relocalization". In *British Machine Vision Conf. (BMVC)*, York, UK, 2018. 1

[12] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. "VidLoc: A Deep Spatio-Temporal Model for 6-DoF Video-Clip Relocalization". In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2652–2660, Honolulu, HI, 2017. 3

[13] Gabriele Costante and Thomas Alessandro Ciarfuglia. "LS-VO: Learning Dense Optical Subspace for Robust Visual Odometry Estimation". In *Robotics and Automation Letters*, volume 3, pages 1735–1742, July 2018. 3

[14] Gabriele Costante, Michele Mancini, Paolo Valigi, and Thomas A. Ciarfuglia. "Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation". In *Robotics and Automation Letters*, volume 1, Jan. 2016. 3

[15] Mingyu Ding, Zhe Wang, Jiankai Sun, Jianping Shi, and Ping Luo. "CamNet: Coarse-to-Fine Retrieval for Camera Re-Localization". In *Intl. Conf. on Computer Vision (ICCV)*, pages 2871–2880, Seoul, South Korea, 2019. 2

[16] Nan-Duong Duong, Amine Kacete, Catherine Sodarie, Pierre-Yves Richard, and Jérôme Royan. "xyzNet: Towards Machine Learning Camera Relocalization by Using a Scene Coordinate Prediction Network". In *Intl. Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pages 258–263, Munich, Germany, 2018. 1

[17] Ryan C. DuToit, Joel A. Hesch, Esha D. Nerurkar, and Stergios I. Roumeliotis. "Consistent map-based 3D localization on mobile devices". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 6253–6260, Singapore, Singapore, 2017. 2

[18] Tobias Feigl, Andreas Porada, Steve Steiner, Christoffer Löfller, Christopher Mutschler, and Michael Philippse. "Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-Scale Industry Environments". In *Intl. Conf. on Computer Graphics Theory and Applications*, Jan. 2020. 2

[19] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philipp Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. "FlowNet: Learning Optical Flow with Convolutional Networks". In *Intl. Conf. on Computer Vision (ICCV)*, pages 2758–2766, Santiago de Chile, Chile, 2015. 5, 8

[20] Yarin Gal and Zoubin Ghahramani. "Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference". In *arXiv preprint arXiv:1506.02158*, 2016. 2

[21] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? The Kitti vision benchmark suite". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, Providence, RI, 2012. 5

[22] Marcel Geppert, Peidong Liu, Zhaopeng Cui, Marc Pollefeys, and Torsten Sattler. "Efficient 2D-3D Matching for Multi-Camera Visual Localization". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 5972–5978, Montreal, Canada, 2019. 2

[23] Abner Guzman-Rivera, Pushmeet Kohli, Ben Glocker, Jamie Shotton, Toby Sharp, Andrew Fitzgibbon, and Shahram Izadi. "Multi-output Learning for Camera Relocalization". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1114–1121, Columbus, OH, 2014. 1

[24] Qiang Hao, Rui Cai, Zhiwei Li, Lei Zhang, Yanwei Pang, and Feng Wu. "3D visual phrases for landmark recognition". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3594–3601, Providence, RI, 2012. 2

[25] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1647–1655, Honolulu, HI, 2017. 2, 3, 4, 5, 8

[26] Ganesh Iyer, J. Krishna Murthy, Gunshi Gupta, K. Madhava Krishna, and Liam Paull. "Geometric Consistency for Self-Supervised End-to-End Visual Odometry". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops*, Salt Lake City, UT, 2018. [1](#)

[27] Giyeong Jang, Sungho Lee, and Inso Kweon. "Color landmark based self-localization for indoor mobile robots". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 1037–1042, Washington, DC, 2002. [1](#)

[28] Eagle S. Jones and Stefano Soatto. "Visual-Inertial Navigation, Mapping and Localization: A Scalable Real-Time Causal Approach". In *Intl. Journal of Robotics Research*, volume 30, pages 407–430, 2011. [2](#)

[29] Anton Kasyanov, Francis Engelmann, Jörg Stückler, and Bastian Leibe. Keyframe-based visual-inertial online slam with relocation. In *Proc. Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 6662–6669, Vancouver, Canada, 2017. [2](#)

[30] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proc. Intl. Workshop on Augmented Reality (IWAR)*, pages 85–94, San Francisco, CA, 1999. [2](#)

[31] Alex Kendall and Roberto Cipolla. "Modelling Uncertainty in Deep Learning for Camera Relocalization". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 4762–4769, Stockholm, Sweden, 2016. [1, 2](#)

[32] Alex Kendall and Roberto Cipolla. "Geometric Loss Functions for Camera Pose Regression with Deep Learning". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 6555–6564, Honolulu, HI, 2017. [2, 3, 8](#)

[33] Alex Kendall, Matthew Grimes, and Roberto Cipolla. "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization". In *Intl. Conf. on Computer Vision (ICCV)*, pages 2938–2946, Santiago de Chile, Chile, 2015. [1, 2, 3, 5, 6, 7](#)

[34] Christian Kerl, Jürgen Sturm, and Daniel Cremers. Dense visual SLAM for RGB-d cameras. In *Proc. Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2100–2106, Tokyo, Japan, 2013. [2](#)

[35] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Intl. Workshop on Augmented Reality (ISMAR)*, pages 1–10, Nara, Japan, 2007. [2](#)

[36] Robin Kreuzig, Matthias Ochs, and Rudolf Mester. "DistanceNet: Estimating Traveled Distance From Monocular Images Using a Recurrent Convolutional Neural Network". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops*, Long Beach, CA, 2019. [1](#)

[37] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. "Camera Relocalization by Computing Pairwise Relative Poses Using Convolutional Neural Network". In *Intl. Conf. on Computer Vision Workshop (ICCVW)*, pages 920–929, Venice, Italy, 2017. [2, 5](#)

[38] Sooyong Lee and Jae-Bok Song. "Mobile robot localization using infrared light reflecting landmarks". In *Intl. Conf. Control, Automation and Systems*, pages 674–677, Seoul, South Korea, 2007. [1](#)

[39] Peiliang Li, Tong Qin, Botao Hu, Fengyuan Zhu, and Shaojie Shen. Monocular visual-inertial state estimation for mobile augmented reality. In *Proc. Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 11–21, Vancouver, Canada, 2017. [2](#)

[40] Wenbin Li, Sajad Saeedi, John McCormac, Ronald Clark, Dimos Tsoumanikas, Qing Ye, Yuzhong Huang, Rui Tang, and Stefan Leutenegger. Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. *arXiv preprint arXiv:1809.00716*, 18(17), 2018. [2](#)

[41] Yunpeng Li, Noah, Noah Snavely, Dan Huttenlocher, and Pascal Fua. "Worldwide Pose Estimation Using 3D Point Clouds". In *Europ. Conf. on Computer Vision (ECCV)*, Oct. 2012. [2](#)

[42] Yimin Lin, Zhaoxiang Liu, Jianfeng Huang, Chaopeng Wang, Guoguang Du, Jinqiang Bai, Shiguo Lian, and Bill Huang. "Deep Global-Relative Networks for End-to-End 6-DoF Visual Localization and Odometry". In *Pacific Rim Intl. Conf. Artificial Intelligence (PRICAI)*, pages 454–467, Yanuca Island, Cuvu, Fiji, 2019. [1, 3](#)

[43] Haomin Liu, Mingyu Chen, Guofeng Zhang, Hujun Bao, and Yingze Bao. Ice-ba: Incremental, consistent and efficient bundle adjustment for visual-inertial slam. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1974–1982, Salt Lake City, Utah, 2018. [2](#)

[44] Christoffer Löffler, Sascha Riechel, Janina Fischer, and Christopher Mutschler. "Evaluation Criteria for Inside-Out Indoor Positioning Systems Based on Machine Learning". In *Intl. Conf. on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, Nantes, France, Sept. 2018. [1, 5, 7](#)

[45] David G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In *Intl. Journal of Computer Vision*, volume 60(2), pages 91—110, 2004. [2](#)

[46] Simon Lynen, Torsten Sattler, Michael Bosse, Joel A. Hesch, Marc Pollefeys, and Roland Siegwart. "Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization". In *Robotics: Science and Systems*, 2015. [2](#)

[47] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. "1 Year, 1000km: The Oxford RobotCar Dataset". In *Intl. Journal of Robotics Research (IJRR)*, pages 3–15, 2016. [5, 8](#)

[48] Syaiful Mansur, Muhammad Habib, Gilang Nugraha Putu Pratama, Adha Imam Cahyadi, and Igi Ardiyanto. Real Time Monocular Visual Odometry using Optical Flow: Study on Navigation of Quadrotora's UAV. In *Intl. Conf. on Science and Technology - Computer (ICST)*, pages 122–126, Yogyakarta, Indonesia, 2017. [3](#)

[49] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: A hands-on survey. *Trans. Visualization and Computer Graphics*, 22(12):2633–2651, 2016. [2](#)

[50] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. "Image-Based Localization Using Hourglass Networks". In *Intl. Conf. on Computer Vision Workshop (ICCVW)*, pages 870–877, Venice, Italy, 2017. [1](#)

[51] Lili Meng, Jianhui Chen, Frederick Tung, James J. Little, Julien Valentin, and Clarence W. da Silva. "Backtracking

Regression Forests for Accurate Camera Relocalization". In *Proc. Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 6886–6893, Vancouver, BC, 2017. 1

[52] Sven Middelberg, Torsten Sattler, Ole Untzelmann, and Leif Kobbelt. "Scalable 6-DOF Localization on Mobile Devices". In *Europ. Conf. on Computer Vision (ECCV)*, 2014. 2

[53] Peter Muller and Andreas Savakis. "Flowdometry: An Optical Flow and Deep Learning Based Approach to Visual Odometry". In *Winter Conf. on Applications of Computer Vision (WACV)*, pages 624–631, Santa Rosa, CA, 2017. 3

[54] Raul Mur-Artal and Juan D. Tardos. "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras". *Trans. Robotics*, 33(5):1255–1262, 2017. 2

[55] Raúl Mur-Artal and Juan D. Tardós. "Visual-Inertial Monocular SLAM With Map Reuse". In *Robotics and Automation Letters*, volume 2, pages 796–803, Apr. 2017. 2

[56] Tayyab Naseer and Wolfram Burgard. "Deep Regression for Monocular Camera-based DoF Global Localization in Outdoor Environments". In *Proc. Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1525–1530, Vancouver, BC, 2017. 2

[57] Riccardo Palmarini, John Ahmet Erkoyuncu, and Rajkumar Roy. "An innovative process to select augmented reality (AR) technology for maintenance". In *Proc. Intl. Conf. on Manufacturing Systems (CIRP)*, volume 59, pages 23–28, Taichung, Taiwan, 2017. 2

[58] Mitesh Patel, Brendan Emery, and Yan-Ying Chen. "ContextualNet: Exploiting Contextual Information using LSTMs to Improve Image-based Localization". In *Intl. Conf. Robotics and Automation (ICRA)*, 2018. 2, 3, 8

[59] Noha Radwan, Abhinav Valada, and Wolfram Burgard. "VLocNet++: Deep Multitask Learning for Semantic Visual Localization and Odometry". *Robotics and Automation Letters*, 3(4):4407–4414, 2018. 1, 3, 5, 8

[60] Soham Saha, Girish Varma, and C.V.Jawahar. "Improved Visual Relocalization by Discovering Anchor Points". In *arXiv preprint arXiv:1811.04370*, Nov. 2018. 2

[61] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. "Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization". In *Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 39(9), pages 1744–1756, Sept. 2017. 2

[62] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5

[63] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixé. "Understanding the Limitations of CNN-based Absolute Camera Pose Regression". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3302–3312, Long Beach, CA, 2019. 1, 2

[64] Thomas Schneider, Marcin Dymczyk, Marius Fehr, Kevin Egger, Simon Lynen, Igor Gilitschenskiz, and Roland Siegwart. "maplab: An Open Framework for Research in Visual-inertial Mapping and Localization". In *Robotics and Automation Letters*, volume 3, pages 1418–1425, July 2018. 2

[65] Soroush Seifi and Tinne Tuytelaars. "How to improve CNN-based 6-DoF Camera Pose Estimation". In *Intl. Conf. on Computer Vision (ICCV)*, 2019. 2, 3, 8

[66] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2930–2937, Portland, OR, 2013. 1, 2, 5, 6, 7, 8

[67] Chris Sweeney, Victor Fragoso, Tobias H. Höllerer, Matthew Tur, and kMatthew Turk. "Large Scale SfM with the Distributed Camera Model". In *arXiv preprint arXiv:1607.03949*, July 2016. 2, 8

[68] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, Boston, MA, 2015. 3, 8

[69] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. "Visual SLAM algorithms: a survey from 2010 to 2016". *Trans. Computer Vision and Applications*, 9(1):452–461, 2017. 2

[70] Takahiro Terashima and Osamu Hasegawa. "A visual-SLAM for first person vision and mobile robots". In *Proc. Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 73–76, Vancouver, Canada, 2017. 2

[71] Akihiko Torii, Relja Arandjelović, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. "2/7 place recognition by view synthesis". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1808–1817, Boston, MA, 2015. 2, 8

[72] Abhinav Valada, Noha Radwan, and Wolfram Burgard. "Deep Auxiliary Learning for Visual Localization and Odometry". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 6939–6946, Brisbane, Australia, 2018. 1, 3, 8

[73] Julien Valentin, Angela Dai, Matthias Niessner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. "Learning to Navigate the Energy Landscape". In *Intl. Conf. on 3D Vision (3DV)*, Oct. 2016. 5

[74] Julien Valentin, Matthias Niessner, Jamie Shotton, Andrew Fitzgibbon, Shahram Izadi, and Philip Torr. "Exploiting uncertainty in regression forests for accurate camera relocalization". In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4400–4408, Boston, MA, 2015. 1

[75] Reid Vassallo, Adam Rankin, Elvis C. S. Chen, and Terry M. Peters. "Hologram stability evaluation for microsoft (r) hololens tm". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 3–14, Marina Bay Sands, Singapur, 2017. 2

[76] Florian Walch, Daniel Cremers, Sebastian Hilsenbeck, Caner Hazirbas, and Laura Leal-Taixé. "Deep Learning for Image-Based Localization". Master's thesis, Technische Universität München, Department of Informatics, Semantic Scholar, Munich, Germany, 2016. 2

[77] Florian Walch, Caner Hazirbas, Laura Leal-Taixé, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. "Image-Based Localization Using LSTMs for Structured Feature

Correlation". In *Intl. Conf. on Computer Vision (ICCV)*, pages 627–637, Venice, Italy, 2017. 1, 2, 4, 5, 6, 7, 8

[78] Junqiu Wang, Hongbin Zha, and Roberto Cipolla. "Coarse-to-Fine Vision-Based Localization by Indexing Scale-Invariant Features". In *Trans. on Systems, Man, and Cybernetics*, volume 36(2), pages 413–422, Apr. 2006. 2

[79] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. "DeepVO: Towards end-to-end Visual Odometry with deep Recurrent Convolutional Neural Networks". In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 2043–2050, Singapore, Singapore, 2017. 2, 8

[80] Qi Zhao, Fangmin Li, and Xinhua Liu. "Real-time Visual Odometry based on Optical Flow and Depth Learning". In *Intl. Conf. on Measuring Technology and Mechatronics Automation (ICMTMA)*, pages 239–242, Changsha, China, 2018. 3

Chapter 11

Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

Contributing Article

Felix Ott, Nisha Lakshmana Raichur, David Rügamer, Tobias Feigl, Heiko Neumann, Bernd Bischl, and Christopher Mutschler. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression. *arXiv preprint arXiv:2208.00919 v3 [cs.CV]*, August 2023.

Publisher Website

Paper available at: <https://arxiv.org/abs/2208.00919>

Copyright License

This work is licensed under a Creative Commons Attribution International 4.0 License (<https://creativecommons.org/licenses/by/4.0/>). © Copyright held by the owner/author(s).

Author Contributions¹⁵

Felix Ott and Nisha Lakshmana Raichur contributed equally to this paper. Felix Ott is the corresponding author. Felix Ott contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the

website), writing of the original draft, reviewing and editing, and visualization. Nisha Lakshmana Raichur contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording and data management), writing of the original draft, reviewing and editing, and visualization. David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Tobias Feigl contributed with writing (i.e., review and editing). Heiko Neumann contributed with writing (i.e., review and editing) and supervision. Bernd Bischl contributed with supervision. Christopher Mutschler contributed with the computing resources, writing (i.e., review and editing), supervision, and funding acquisition by the research program Human-Computer-Interaction as well as the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

Statement on Differences between Master’s Thesis and Paper

The contributing paper is founded on Nisha Lakshmana Raichur’s master’s thesis titled “Image-/IMU-based Deep Multimodal Information Fusion for Pedestrian Self-Localization” which was submitted to the University of Ulm on April 15, 2021. The thesis was supervised by Felix Ott, Friedhelm Schwenker, and Heiko Neumann. The primary objective of the master’s thesis was to propose base models that could independently process images and IMU sensor data to estimate absolute and relative poses, respectively. Furthermore, various multi-modal fusion techniques were employed to integrate the outputs of absolute and relative pose regression models. The efficacy of the proposed methods was evaluated on three datasets: the state-of-the-art aerial EuRoC MAV (Burri et al., 2016) dataset, the hand-held PennCOSYVIO (Pfrommer et al., 2017) dataset, and a hand-held dataset (one part of the Industry scenario #4 dataset) recorded at the L.I.N.K. test and application center at the Fraunhofer IIS in Nürnberg, Germany. Drawing from the aforementioned research, the authors of the contributing paper conducted a comprehensive assessment of visual-inertial deep multimodal fusion for estimating the absolute and relative pose. The contributing paper provides a thorough investigation of state-of-the-art pose regression techniques and datasets beyond what was covered in the master’s thesis. The authors generated new figures and refined the method descriptions. Furthermore, all models were retrained with pre-selected model parameters. The paper also includes an evaluation of network mask performance. Figure 38 to Figure 41 in the appendix of the contributing paper are taken from the master’s thesis. The newly introduced IndustryVI datasets are presented along this paper.

Datasets

This paper uses the publicly available EuRoC MAV and PennCOSYVIO datasets. Along this paper, the IndustryVI datasets were published and are available at:

<https://www.iis.fraunhofer.de/de/ff/1v/lok/opt1/warehouse.html>

https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets

Statement about Recent, Related Research¹⁷

In the contributing paper (Ott et al., 2023b), we conduct a study where we combine APR and RPR models and evaluate various sensor fusion techniques. Xu et al. (2022) reviewed conventional methods for estimating camera pose based on structure, as well as APR and RPR methods. They also proposed improvements to these methods, including neural network structures and loss functions, with the aim of inspiring further advancements. By incorporating advancements made in the sub-methods of APR or RPR, the sub-modules of the sensor fusion architecture employed in the contributing paper (Ott et al., 2023b) could be improved, resulting in better pose regression results. In Zhuang & Chandraker (2021), the authors combine APR with solutions derived from geometric solvers, leveraging their complementary benefits in a manner that can be learned. They achieved this by probabilistically fusing the uncertainty of the APR network, guided explicitly by the geometric uncertainty, thereby taking into account the geometric solution in relation to the network's prediction. Similar, Brieger et al. (2022) combined visual and inertial data for global navigation satellite system (GNSS) interference detection, akin to the process of visual-inertial fusion for pose regression. The authors combine a ResNet architecture that is based on visual features with a time-series Transformer network that is based on time-series features. They evaluate the performance of late fusion, intermediate fusion using the multi-modal transfer module (MMTM) by Joze et al. (2020), and soft fusion (Chen et al., 2019). The results show that while the late fusion approach achieves the highest classification accuracy for GNSS interference detection, the MMTM and soft fusion approaches outperform the late fusion model for pose regression in the contributing paper (Ott et al., 2023b). This suggests that the effectiveness of sensor fusion techniques depends on the specific application and data being used.

Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

Felix Ott*, Nisha Lakshmana Raichur*, David Rügamer, Tobias Feigl, Heiko Neumann, Bernd Bischl, Christopher Mutschler

Abstract—Visual-inertial localization is a key problem in computer vision and robotics applications such as virtual reality, self-driving cars, and aerial vehicles. The goal is to estimate an accurate pose of an object when either the environment or the dynamics are known. Absolute pose regression (APR) techniques directly regress the absolute pose from an image input in a known scene using convolutional and spatio-temporal networks. Odometry methods perform relative pose regression (RPR) that predicts the relative pose from a known object dynamic (visual or inertial inputs). The localization task can be improved by retrieving information from both data sources for a cross-modal setup, which is a challenging problem due to contradictory tasks. In this work, we conduct a benchmark to evaluate deep multimodal fusion based on pose graph optimization and attention networks. Auxiliary and Bayesian learning are utilized for the APR task. We show accuracy improvements for the APR-RPR task and for the RPR-RPR task for aerial vehicles and hand-held devices. We conduct experiments on the EuRoC MAV and PennCOSYVIO datasets and record and evaluate a novel industry dataset.¹

Index Terms—camera localization, inertial odometry, visual odometry, multimodal fusion, attention networks, multi-task learning, auxiliary learning, Bayesian networks.

I. INTRODUCTION

LOCATION is important for intelligent systems such as virtual and mixed reality, increasingly deployed in areas of tourism, education, and entertainment [9], [51], [130]. Accurately localizing objects is key to many path planning applications to determine future movements [19], [102], [143] of mobile objects, e.g., robots or micro aerial vehicles (MAVs) [147], [150]. This allows for monitoring and optimizing workflows as well as tracking goods for automated inventory management in real-time. A prerequisite for success is a highly accurate pose recognition (i.e., position and orientation) of the object. Environments in which such objects are typically used include large warehouses, factory buildings, and shopping centers. Localization solutions often use a combination of

LiDAR-, radio-, or radar-based systems [64], [77], which, however, either require additional infrastructure or are costly in their operation. An alternative approach is an optical pose estimation. The accuracy of the pose estimation depends to a large extent on suitable invariance properties of the available features such that these features can be reliably detected [88]. For image-based localization, additional contextual information such as 3D models of the scene or pre-recorded landmark databases can be used when the environment is known. This potentially improves the pose accuracy but also increases the system's complexity. State-of-the-art mobile setups often use cheap sensors such as an inertial measurement unit (IMU) [95], from which different motion dynamics (such as the slow movement of a robot or fast walking and rotation of a human) can be learned in advance. The goal of approaches using both sensors simultaneously is to utilize the advantages of both image and inertial data to improve the self-localization.

Absolute pose regression (APR) techniques directly regress the absolute six degree-of-freedom (6DoF) pose from images (APR_V) and have become popular in recent years. These techniques are based on convolutional neural networks (CNNs) [35], [70], [72], [76], [94], [104], [117], [122], [144] in combination with recurrent neural networks (RNNs) [28], [109], [111], [124], [140]. However, they do not achieve the same level of pose accuracy as 3D structure-based methods [122]. On the other hand, visual odometry (VO) techniques predict the 6DoF relative pose between image pairs of consecutive time steps. Recently, end-to-end approaches utilize CNNs in combination with RNNs for relative pose regression (RPR_V) [29]–[31], [62], [75], [85], [92], [99], [142], [153]. Another approach is inertial odometry (IO), which estimates the 6DoF relative pose from IMUs of consecutive time steps. Classical (non ML-) approaches are [23], [38], [58]. In the context of inertial RPR (RPR_I), recent techniques predict the relative pose with CNNs or RNNs [24], [29], [36].

*F. Ott and N. L. Raichur contributed equally.

F. Ott, N. L. Raichur, T. Feigl, and C. Mutschler are with the Fraunhofer IIS, Fraunhofer Institute for Integrated Circuits IIS, Nuremberg, Germany. E-mail: {felix.ott, nisha.lakshmana.raichur, tobias.feigl, christopher.mutschler}@iis.fraunhofer.de

F. Ott, D. Rügamer, and B. Bischl are with the Statistical Learning and Data Science Chair, LMU Munich, Munich, Germany, and with the Munich Center for Machine Learning (MCML), Munich, Germany. E-mail: {david.ruegamer, bernd.bischl}@stat.uni-muenchen.de

D. Rügamer is with the Technical University of Dortmund, Germany

H. Neumann is with the Institute of Neural Information Processing, Ulm University, Ulm, Germany. E-mail: heiko.neumann@uni-ulm.de

¹Datasets: https://gitlab.cc-asf.fraunhofer.de/ottf/industry_datasets

Odometry techniques typically suffer from accumulated errors and high drifting errors. For IO systems, the method continually integrates acceleration and angular velocities with respect to time to calculate the pose changes [38]. Measurement errors – even if small individually – accumulate over time and lead to long-term drifts, i.e., due to temperature changes or loosely placed sensors [82]. The drifting error of VO techniques arises from fast movement changes and image blur that is handled by loop closure [20]. Although VO has made remarkable progress over the last decade, it still

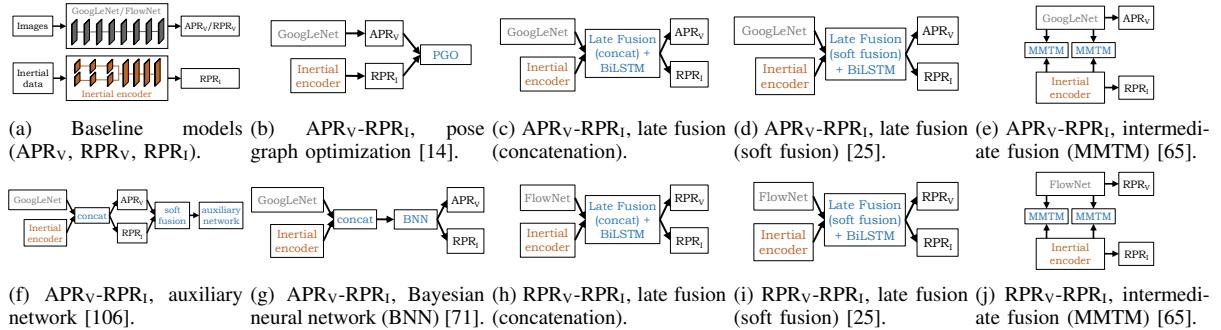


Fig. 1: Structures of the benchmark methods for visual-inertial fusion for absolute (APR) and relative (RPR) pose regression. The baselines APR_v (PoseNet [72]), RPR_v (FlowNet [37]), and RPR_i (IMUNet [36]) models are fused for a common representation.

suffers greatly from scaling errors [68], [74], [80], [81], [86], [102]. While APR_v is highly accurate by relying on distinct observed features in the environment (e.g., texture-rich scenes with perfect illumination), its accuracy is largely degraded by appearance changes caused by intermittent occlusions such as moving objects, photometric calibration, low-light conditions, and illumination changes [149]. Recent techniques tackle this problem with a fusion of camera and IMU sensors [25], [29], [85], [109], [117].

As both sensor types have different advantages in different situations, visual and inertial data can be used simultaneously to achieve a highly accurate pose in long-term use. The goal of fusion approaches is to reduce the drifting error of odometry techniques by utilizing the absolute pose and to reduce the error of APR in texture-less environments utilizing the relative pose. Classical fusion methods of visual-inertial (VI) systems are based on Bayesian filtering [7], [97], [127] or on optimization-based methods [127]. Traditionally, these methods rely on 3D maps and local features [60], [83], [120]. However, naively using all the features before fusion will lead to unreliable state estimation, incorrect feature extraction, or a matching that cripples the entire system [151]. Hybrid methods combine geometric and ML approaches to predict the 3D position of each pixel in world coordinates [11], [125]. With more computing power, VI SLAM partly resolves the scale ambiguity to provide motion cues without visual features [68], [86], [133] and to make tracking more robust [73]. Multiple works combine global localization in a scene with VO or IO [22], [40], [89], [102], [103], [108].

For multimodal learning, several streams are constructed to optimally perform individual tasks at different levels – i.e., early, intermediate, and late fusion. Although, studies suggest the use of intermediate fusion [67], [110], late fusion is still the predominant method due to practical reasons. In the field of VI-based learning, intermediate features of the encoders have unaligned spatial dimensions, which make intermediate fusion more challenging [25], [29], [85], [109], [117]. Commonly, 1D feature vectors from each unimodal stream are concatenated and an attention mechanism chooses the best expert for each input signal [25], [65], or dense networks are used for hierarchical joint feature learning [56]. With multi-task

learning (MTL), a model learns multiple tasks simultaneously [26], [87], [129] with a shared representation that contains the mutual concepts between multiple related tasks. In contrast, auxiliary learning models can be trained on the main task of interest with multiple auxiliary tasks [84], [106], [134]. This adds an inductive bias that pushes models to capture meaningful representations and improves generalization. Training APR and RPR networks can be interpreted as MTL with equal tasks, while – in the context of auxiliary learning – APR is the main task and RPR is the auxiliary task. A different field is uncertainty quantification. Estimating the uncertainty in position estimation provides significant insight into the model reliability. One possibility to explain models better is to estimate their aleatoric and epistemic uncertainty [44], [71]. Bayesian methods show robustness to noisy data and provide a practical framework for understanding uncertainty in models [12], [27], [66], [128]. For APR and RPR fusion, the model can learn to rely on the absolute or relative pose prediction dependent on the quantified aleatoric uncertainty.

Contributions. In this work, our main objective is to evaluate a wide range of different, fundamental fusion techniques (see Figure 1), that proved to be effective in different fields, for the VI pose regression problems (APR_v-RPR_i and RPR_v-RPR_i). The issue at hand involves the global and relative pose in order to optimize the global pose, and mitigating the effects of environmental factors on the fusion techniques. (1) We provide an overview of APR_v, RPR_v, and RPR_i methods, and use PoseNet [72], FlowNet [37], and IMUNet [36] as baseline models. Indeed, there are more advanced models that yield a lower localization error in the context of APR and RPR. Instead, we benchmark different fusion techniques and highlight their influence on the pose regression tasks. (2) We apply pose graph optimization (PGO) [50] for absolute pose refinement (see MapNet [14]), and absolute and relative pose fusion. (3) We evaluate attention-based methods for late fusion (concatenation and soft fusion [25]) and intermediate fusion, i.e., multimodal transfer module (MMTM) [65], for a cross-modal feature representation. (4) We utilize non-linear and convolutional auxiliary learning [106] and quantify the aleatoric uncertainty using Bayesian networks [71] to improve the loss of the main APR_v task. (5) We record a large indoor

industrial dataset and benchmark the EuRoC MAV [18], the PennCOSYVIO [112], and our IndustryVI datasets. To further enhance the results, advanced techniques may serve as black box models in place of the baseline models. Instead, our contribution is to provide insights into the role of environmental changes and motion dynamics on the localization task as well as the robustness of the fusion techniques by exploring results on these three different datasets.

The remainder of our paper is organized as follows. We first discuss related work in Section II. Section III introduces our framework and methods. We present a novel dataset in Section IV, before showing experimental results in Section V.

II. RELATED WORK

We address methods for VI self-localization in Section II-A, focus on multimodal fusion techniques in Section II-B, and briefly summarize uncertainty estimation techniques in Section II-C. In Section II-D, we give an overview of datasets.

A. Methods for VI Self-Localization

For self-localization, we separate between odometry methods, and learning-based APR and RPR methods. Odometry methods can be partitioned into VO, IO, and VI odometry, each separated into classical and regression-based techniques. As this paper provides a benchmark for APR and RPR fusion techniques, in this section, we briefly summarize methods that purely rely on SLAM, odometry, RPR, or APR (that build our baseline models). We focus on the combination of the classical VO and IO methods, the combination of RPR_V and RPR_I, and the combination of APR_V and RPR (see Section II-B).

VO & RPR_V. Classical VO methods are based on a dead reckoning system using image features [92], or use point features between pairs of frames from stereo cameras [107]. For an overview, see [93]. DeepVO [142] is one of the first networks that combined CNNs with RNNs (two stacked LSTMs) to model sequential dynamics and relations. While CTCNet [62] predicts relative poses from a CNN+LSTM model, DistanceNet [75] uses a CNN+BiLSTM to estimate traveled distances divided into classes. Several methods exist that use the optical flow (OF) to regress the relative pose from consecutive image pairs – such as Flowdometry [99], [100], ViPR [109], DeepVIO [52], and KFNet [154]. These methods either use FlowNet [37] or FlowNet2 [61]. While LS-VO [30] uses an autoencoder for OF prediction, P-CNN [31] uses the Brox algorithm [17] with a standard CNN. P-CNN+OF [153] combines FlowNet2 and P-CNN. DF-VO [152] proposed a method integrating epipolar geometry from CNNs (single-view depth and OF) and the Perspective-n-Point [45] method. D3VO [148] learns the image depth, models the pixel uncertainties which improves the depth estimation, and predicts the pose with PoseNet. In this paper, we use FlowNetSimple [37] to predict the RPR_V relative pose (see Figure 1a).

IO & RPR_I. Classical IO includes pedestrian dead reckoning (often combined with GPS signals) [5], is based on walk detection and step counting [15], or is designed as a strapdown inertial navigation system [23], [126]. While IONet [24] segments inertial data into independent windows

and learns the relative pose with a CNN+RNN, VINet [29] directly estimates features from IMU data with LSTMs. Silva et al. [36] propose a CNN+BiLSTM model (IMUNet) and evaluate different loss functions (i.e., based on a vector in the spherical coordinate system, or based on a translation vector and a unit quaternion). As the CNN+BiLSTM model [36] yields state-of-the-art results, we use this model as RPR_I baseline (see Figure 1a).

APR. APR methods are based on CNNs for (re-)localization such as GoogLeNet [132] or ResNet [137]. Methods as (Dense) PoseNet [72], BranchNet [104], Hourglass [94], Geometric PoseNet [70] and Bayesian PoseNet [69] are partially insensitive to occlusions, light changes, and motion blur. As PoseNet [72] evolved as a simple and effective APR technique, we use PoseNet as APR_V baseline method (see Figure 1a). [144] dealt with the coupling of orientation and translation by splitting the network into two branches. LSTMs [53] and BiLSTM [49] were utilized to extract the temporal context, e.g., PoseNet+LSTM [140], ViPR [109], ContextualNet [111] and Seifi et al. [124] use LSTMs, while VidLoc [28] uses BiLSTMs. RelocNet [4] learns metrics continuously from global image features through a camera frustum overlap loss. CamNet [35] is a coarse-to-fine retrieval-based model that includes relative pose regression to get close to the best database entry that contains extracted features of images. NNet [76] queries a database for similar images to predict the relative pose between images and RANSAC solves the triangulation to provide a position. The CNN by [13] densely regresses so-called scene coordinates, defining the correspondence between the input image and the 3D scene space. Sattler et al. [122] showed that pose regression is more closely related to pose approximation via image retrieval than to accurate pose estimation via 3D structure by predicting failure cases. [10] showed that learning-based scene coordinate regression outperforms classical feature-based methods. MapNet [14] learns a map representation by geometric constraints that are formulated as loss terms. [59] add a prior guided dropout module before PoseNet with spatial and channel attention modules to guide CNNs to ignore foreground objects. [113] inferred a depth map from a CNN encoder and predicted the pose from the most similar image with nearest neighbor indexing. AtLoc [141] consists of a visual encoder that extracts features and an attention module that computed the attention and re-weights features.

B. Multimodal Fusion for Self-Localization

Classical VI Odometry. Classical methods for sensor fusion are the (extended) Kalman filter (KF) [48], [91] or pose graphs [32]. [58] use a trifocal tensor geometry between three images without estimating the 3D position of feature points (without reconstructing the environment). The poses are refined with a multi-state KF in combination with a RANSAC algorithm. They transform the camera frame w.r.t. the IMU frame. VINS-Mono [115] is a nonlinear optimization-based method for VI odometry by fusing pre-integrated IMU measurements and feature observations that merge maps by PGO [50].

RPR-based VI Odometry. DeepVIO [52] learns the OF from consecutive images and the relative pose from an inertial-

TABLE I: Overview of visual and inertial datasets and its sensors and ground truth properties.

Dataset	Ref.	Year	Environment	Carrier	Sensors	Ground truth
TUM RGB-D	[131]	2012	Indoors	Pioneer Robot	Cam: 2 stereo RGB-D (30 Hz), IMU: 1 acc. (500 Hz)	Motion capture (300 Hz)
KITTI	[47]	2012	Outdoors	Car	Cam: 1 stereo RGB/gray, IMU: 1 acc./gyr., laser	INS/GNSS (10 Hz)
Microsoft 7-Scenes	[125]	2013	Indoors	Handheld	Cam: RGB-D	KinectFusion
Málaga Urban	[6]	2014	Outdoors	Car	Cam: 1 stereo RGB, IMU: 1 acc., 2 gyr.	GPS (1 Hz)
Cambridge Landmarks	[72]	2015	Outdoors	Handheld, urban	Cam: Google LG Nexus 5 smartphone (2 Hz)	From SfM
UMich NCLT	[21]	2016	In-/outdoors	Segway	Cam: 6 omnid. RGB (5 Hz), IMU: 1 acc., 2 gyr., laser	GPS/IMU/laser
EuRoC MAV	[18]	2016	Indoors	MAV hexacopter	Cam: 1 stereo gray (20 Hz), IMU: 1 acc./gyr. (200 Hz)	MoCap Laser (20 Hz)
12-Scenes	[135]	2016	Indoors	Handheld	Cam: 1 RGB, 1 RGB-D	VoxelHashing
Oxford RobotCar	[90]	2016	Outdoors	RobotCar	Cam: Bumblebee XB3	GPS, IMU (5 Hz)
PennCOSYVIO	[112]	2016	In-/outdoors	Handheld	Cam: 4 RGB, 1 stereo, 1 fisheye, IMU: 1 acc./gyr.	Visual tags (30 Hz)
Zurich Urban MAV	[2]	2017	Outdoors	MAV	Cam: 1 RGB (30 Hz), IMU: 1 acc./gyr. (10 Hz)	Pix4D visual pose
Aalto University	[76]	2017	Indoors	Handheld	Cam: iPhone 6S smartphone	Google Project Tango's
TUM-LSI	[140]	2017	Indoors	NavVis system	Cam: 6 Panasonic wide-angle from NavVis M3 system	Hokuyo laser (SLAM)
Warehouse	[88]	2018	Indoors	Pos. system	Cam: 8 60° RGB	Nikon iGPS
DeepLoc	[117]	2018	Outdoors	Robot platform	Cam: RGB-D ZED stereo (20 Hz), IMU: XSens, LiDARs	GPS
TUM VI	[123]	2018	In-/outdoors	Handheld	Cam: 1 stereo gray (20 Hz), IMU: 1 acc./gyr. (200 Hz)	Partial motion
CMU Seasons	[121]	2018	Outdoors	Car, suburban	Cam: 45° forward/left and forward/right	SIFT, BA
Industry	[109]	2020	Indoors	Forklift, pos. sys.	Cam: 4 170° RGB on fork., 3 170° RGB on pos. sys.	Qualisys (140 Hz)
UMA-VI	[156]	2020	In-/outdoors	Handheld	Cam: RGB (12.5 Hz), gray (25 Hz), IMU: acc./gyr. (250 Hz)	Visual pose (partial)
IndustryVI	ours	2022	Indoors	Handheld	Cam: OrbbeC3D (23 Hz), IMU: 1 acc./gyr./mag. (140 Hz)	Qualisys (140 Hz)

based network, fused with fully connected layers for VI ego-motion. This is supported by a network with stereoscopic image inputs. SelfVIO [1] combined VO and IO networks with an adaptive fusion model that concatenates features of both networks, selects features and predicts the absolute pose with an LSTM. Similarly, the selective sensor fusion (SSF) approach by [25] extracts features from image and IMU data, uses a soft or hard (based on Gumbel softmax) fusion approach to select features and regresses the pose with an LSTM. We use the soft fusion approach of SSF to combine APR_V-RPR_I (see Figure 1d) and RPR_V-RPR_I (see Figure 1i).

APR & RPR Fusion. Learning-based methods are based on APR or RPR networks. VI-DSO [138] jointly estimates camera poses and sparse scene geometry by minimizing the photometric and the IMU measurement error in a combined energy functional. The loss formulation of LM-Reloc [139] is inspired by the Levenberg-Marquardt algorithm, such that the learned features significantly improve the robustness of direct image alignment. Additionally, their network performs RPR to bootstrap the direct image alignment. VINet [29] incorporates relative features from an LSTMs inertial encoder with absolute features from a visual encoder by concatenation. ViPR [109] concatenates relative poses (from OF) and absolute poses to refine the absolute poses with an LSTM network. MapNet+PGO [14] uses PGO [50] to refine predicted poses from absolute and relative pose predictions (we use PGO in Figure 1b). VLocNet [134] estimates a global pose and combines it with VO. To further improve the (re)-localization accuracy, VLocNet++ [117] uses features from a semantic segmentation.² RCNN [85] fuses relative and global networks – while the relative sub-networks smooth the VO trajectory, the global sub-networks avoid the drift problem. Their cross transformation constraints represent the temporal geometric consistency of consecutive frames. We use concatenation as a baseline fusion technique (see Figure 1c and 1h).

²Public code is not available for VLocNet. Re-implementations lead to subpar performance [33], but close to MapNet [14].

C. Uncertainty Estimation for Multimodal Fusion

KFNet [154] extends the scene coordinate regression problem to the time domain based on KF, OF, and Bayesian learning. [119], [145] show that the training can allow uncertainty predictions through a Gaussian density loss in combination with a KF. ToDayGAN [3] is a generative network that alters nighttime driving images to a more useful daytime representation captured from two trajectories of the same area in both day and night. The dropout module by [59] enables the pose regressor to output multiple hypotheses from which the uncertainty of pose estimates can be quantified. CoordiNet [96] predicts the pose and uncertainty from a single loss function for visual relocalization that are fused with a KF to embed the scene geometry. [16] utilized MMTM modules to fuse features derived from images (ResNet) and features extracted from time-series data (TS Transformer) while assessing Monte-Carlo dropout. This approach bears similarity to our own setup, demonstrating that MMTM can be employed in diverse areas of study.

D. Datasets

Each application covers different characteristics that have to be represented by the dataset, i.e., properties of the environment (small or large scale, features), properties of the object (movement patterns such as direction, velocity, and acceleration), and size of the dataset. We provide a benchmark on different datasets to evaluate the performance of models in certain scenarios. A dataset can be classified with the following properties: Indoor or outdoor environment, small- or large-scale environment, a high accuracy (< 1cm) of the ground truth trajectory, availability of datasets, same spatial range of training and test trajectories, and whether the training dataset allows a generalized training.

Table I summarizes all VI self-localization datasets and their characteristics. As the TUM RGB-D [131], Microsoft 7-Scenes [125], Cambridge Landmarks [72], 12-Scenes [135], Aalto University [76], DeepLoc [117], TUM-LSI [140], Warehouse [88] and Industry [109] datasets contain only images,

we cannot make use of these datasets for our VI multimodal setup. The KITTI [46], [47], Málaga Urban [6], Oxford RobotCar [90], and UMICH NCLT [21] datasets contain IMU and LiDAR data, but cannot be used for APR as odometry is the main task. The Aachen day-night, CMU seasons, and RobotCar seasons datasets [121] address viewing conditions such as weather and seasonal variations and day-night changes for visual localization. Similarly, the Zurich Urban MAV [2] dataset addresses only evaluations for VO and SLAM. UMA-VI [156] is a handheld indoor and outdoor dataset with VI data but contains only partial ground truth poses. Also, the ground truth systems of TUM VI [123] cover only a single room, and hence, longer trajectories do not cover ground truth poses.

The EuRoC MAV [18] dataset contains VI data recorded with an MAV in indoor small-scale environments and is suitable for our APR-RPR benchmark. In contrast, PennCOSYVIO [112] was recorded handheld in a large-scale outdoor and indoor environment with VI sensors. As this dataset does not allow evaluations across various movement patterns, we record the IndustryVI dataset in a challenging large-scale indoor industrial environment with ground truth accuracies below 1mm. We let two persons walk with a handheld device with various movement patterns. Having two different persons, in particular, allows an evaluation of the generalizability between different walking styles. For more information, see Section IV. Hence, we use the EuRoC MAV, the PennCOSYVIO, and our IndustryVI datasets to benchmark different fusion models on indoor and outdoor applications.

III. METHODOLOGY

In this section, we present different deep multimodal fusion techniques for odometry-aided APR and VI odometry. Section III-A presents the baseline models for APR_V , RPR_I , and RPR_V . We describe PGO for absolute pose refinement in Section III-B. Section III-C proposes attention-based fusion methods. We use auxiliary learning for APR_V - RPR_I fusion in Section III-D, and use Bayesian neural networks (BNNs) for aleatoric uncertainty estimation in Section III-E. We describe fusion techniques for RPR_V in Section III-F. Table II summarizes the notations.

A. Baseline Models

The baseline of our fusion models is established through the utilization of APR and RPR models. A CNN-based APR model is capable of learning to directly regress the camera pose from a single image or a set of training images in conjunction with their corresponding ground truth poses. In contrast, an RPR model performs 6DoF odometry through the utilization of either inertial data obtained from an IMU (RPR_I) or visual data obtained from a camera (RPR_V).

APR_V. We use PoseNet [72] with time-distribution [109] to predict the absolute positions $\mathbf{p} \in \mathbb{R}^3$ and the absolute orientations $\mathbf{q} \in \mathbb{R}^4$ as illustrated in Figure 2. PoseNet is a CNN architecture that utilizes GoogLeNet [132], which is pre-trained on a huge classification dataset such as ImageNet [34]. PoseNet adds a fully connected (FC) layer of 2,048 units on

TABLE II: Key parameters and their descriptions.

Parameter	Description
APR_V	Absolute pose regression based on visual input
RPR_V	Relative pose regression based on visual input
RPR_I	Relative pose regression based on inertial input
$\mathbf{x} = [\mathbf{p}, \mathbf{q}]$	Absolute pose
$\Delta \mathbf{x} = [\Delta \mathbf{p}, \Delta \mathbf{q}]$	Relative pose
$\mathbf{p} \in \mathbb{R}^3$	Absolute 3D position in Euclidean space
$\mathbf{q} \in \mathbb{R}^4$	Absolute orientation as quaternion
$\Delta \mathbf{p} \in \mathbb{R}^3$	Relative position (translation)
$\Delta \mathbf{q} \in \mathbb{R}^4$	Relative orientation (rotation)
\mathbf{v}_{ij}	Relative pose between predicted poses \mathbf{x}_i and \mathbf{x}_j
\mathcal{L}	Loss function
E_c	Constraint energy
$f(c)$	Prediction function
\mathbf{S}_c	Distance matrix
\mathbf{J}	Jacobian
\mathbf{r}	Residual
\boxplus	Manifold update operations for quaternions
\odot	Element-wise multiplication
D	Dataset of size $ D $
$\mathbf{a}_I, \mathbf{a}_V$	Inertial and visual features
α, β, γ	Hyperparameters for loss weighting
\mathcal{S}	Soft fusion operator
g	Sigmoid function
\mathbf{A}, \mathbf{B}	Feature at any given level of APR or RPR
Z	Latent representation
E_A, E_B	Excitation signals
\mathbf{W}	Weights of the network
s	Log variance

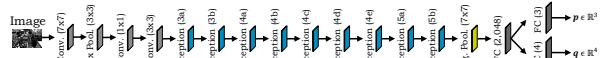


Fig. 2: Structure of PoseNet [72] with a modified GoogLeNet [132] network as the APR_V module with input images I and output position \mathbf{p} and orientation \mathbf{q} .

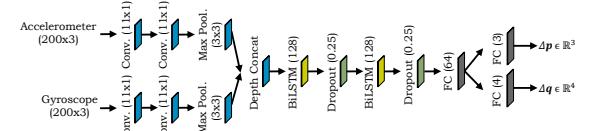


Fig. 3: Structure of the RPR_I module [36] with inertial input data (accelerometer and gyroscope data from 200 timesteps) and output translation $\Delta \mathbf{p}$ and change in orientation $\Delta \mathbf{q}$.

top of the last inception module to form a localization feature vector that may be trained for generalization. In addition, we replace the FC layer and softmax layers of GoogLeNet with two parallel FC layers, each with three and four units. These units are utilized to regress the pose, represented by the $\mathbf{p} = [x, y, z]$ coordinates of the position in Euclidean space and $\mathbf{q} = [w, p, q, r]$ as a quaternion for orientation [70], [155].

RPR_I. Inspired by [36], our RPR_I model is designed based on a CNN combined with two BiLSTM units, as depicted in Figure 3. This design is highly suitable for problems that require the processing of sequential data. The model comprises of 1D-convolutional layers, each with 128 features and a kernel size of 11, which separately process the gyroscope and accelerometer data. After two 1D convolutions, the feature vectors are down-sampled in size through a max-pooling

layer with a size of 3. The outputs of the two separate processing streams are concatenated and fed into a BiLSTM layer, consisting of 128 hidden units. A BiLSTM is utilized to enable the past and future IMU readings to influence the regressed relative pose. To prevent overfitting, a dropout layer with a rate of 25% is added after each BiLSTM layer. Finally, the relative pose $\Delta \mathbf{p} \in \mathbb{R}^3$ and $\Delta \mathbf{q} \in \mathbb{R}^4$ – is regressed through the use of FC layers that take the output from the BiLSTM layers as input.

RPR_V. As the feature encoder in our model, we utilize FlowNetSimple [37] pre-trained on Flying Chairs [37] dataset. This encoder predicts the relative position $\Delta \mathbf{p} \in \mathbb{R}^3$ and orientation $\Delta \mathbf{q} \in \mathbb{R}^4$ from a pair of consecutive monocular images as input. The network is comprised of 9 convolutional layers and is designed to have sufficient capacity to learn the prediction of the OF. The size of the receptive fields gradually decreases from 7×7 to 5×5 , and finally 3×3 . We use the features from the last convolution layer (conv6) as an input to a FC layer consisting of 2,048 units, which forms a localization feature vector. Finally, we add two parallel FC layers, each containing three and four units. These units perform regression of the relative pose represented by $\Delta \mathbf{p}$ and $\Delta \mathbf{q}$.

B. Pose Graph Optimization (PGO)

PGO estimates smooth and globally consistent pose predictions from absolute and relative pose measurements during inference. The method is formulated as a non-convex minimization problem, which can be represented as a graph with vertices corresponding to the estimated global poses edges representing the relative measurements. The objective of PGO is to refine the predicted poses during inference such that the refined poses are close to the actual poses and ensuring agreement between the refined relative poses and the input relative poses [50]. The algorithm utilizes the predicted absolute poses and the relative poses between them as inputs. In the following, we represent the poses

$$\mathbf{p}_i = (t_x, t_y, t_z, q_w, q_p, q_q, q_r) \quad (1)$$

as a vector for translation $\mathbf{t} \in \mathbb{R}^3$ and a vector for orientation represented as quaternion $\mathbf{q} \in \mathbb{R}^4$. To perform PGO for the predicted absolute poses (from the APR_V model), we initially collect all the absolute poses in a single vector \mathbf{z} . We define the objective function, which is the sum of the costs of all the constraints E_c . The constraints can be either for the absolute poses or for the relatives poses (between a pair of absolute poses) for both translation and rotation. The constraint energy

$$E_c(\mathbf{z}) = (f_c(\mathbf{z}) - k_c)^T \mathbf{S}_c (f_c(\mathbf{z}) - k_c) \quad (2)$$

is represented as a quadratic penalty on the difference between $f_c(\mathbf{z})$ and its desired value k_c where $f_c(\mathbf{z})$ is a prediction function that maps the state vector \mathbf{z} to the quantity relevant for constraint c , weighted by distance matrix \mathbf{S}_c (the inverse of the covariance matrix). Equation (1) represents the relative position constraint k_c by having $f_c(\mathbf{z})$ to be the relative position between two poses. We initially linearize f_c around $\bar{\mathbf{z}}$, the current value of the state vector \mathbf{z} , using the substitution $\mathbf{z} = \bar{\mathbf{z}} + \mathbf{x}$, where \mathbf{x} represents the parameter update we solve

for [50]. We take the Cholesky decomposition of the stiffness matrices $\mathbf{S}_c = \mathbf{L}_c \mathbf{L}_c^T$. With

$$f_c(\mathbf{z}) \approx f_c(\bar{\mathbf{z}}) + \frac{\partial f}{\partial \mathbf{z}} \Big|_{\bar{\mathbf{z}}} \mathbf{x}, \quad (3)$$

we get

$$E = \sum_c \left\| \mathbf{L}_c^T \left(f_c(\bar{\mathbf{z}}) + \frac{\partial f}{\partial \mathbf{z}} \Big|_{\bar{\mathbf{z}}} \mathbf{x} - \mathbf{k}_c \right) \right\|^2. \quad (4)$$

Let $\mathbf{J}_c = \mathbf{L}_c^T \frac{\partial f}{\partial \mathbf{z}} \Big|_{\bar{\mathbf{z}}}$ and $\mathbf{r}_c = \mathbf{L}_c^T (k_c - f_c(\bar{\mathbf{z}}))$. We get

$$E = \sum_c \left\| \mathbf{J}_c \mathbf{x} - \mathbf{r}_c \right\|^2 = \left\| \mathbf{J} \mathbf{x} - \mathbf{r} \right\|^2. \quad (5)$$

Stacking the individual Jacobians \mathbf{J} and the residuals \mathbf{r} , we arrive at the least squares problem

$$\Delta \mathbf{z} = \min_{\Delta \mathbf{z}} \left\| \mathbf{J} \Delta \mathbf{z} - \mathbf{r} \right\|^2 \quad (6)$$

to solve for $\Delta \mathbf{z}$. This can be solved by $\Delta \mathbf{z} = (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T \mathbf{r}$. Finally, the predicted absolute pose state vector \mathbf{z} is updated using $\mathbf{z} = \mathbf{z} \boxplus \Delta \mathbf{z}$, where \boxplus represents the manifold update operations for quaternions as described in [14].

PGO during Inference. During inference, the absolute pose predictions and the relative poses between them are used to obtain the optimal absolute poses using PGO. The algorithm runs iteratively utilizing a moving temporal window size of T frames. Suppose the absolute predictions for T frames are $\{\mathbf{p}_i\}_{i=1}^T$ and the relative poses between them are $\{\Delta \mathbf{p}_{ij}\}_{i=1}^{T-1}$, the optimal poses $\{\mathbf{p}^o\}_{i=1}^T$ are solved by minimizing the following cost

$$\mathcal{L}_{\text{PGO}}(\{\mathbf{p}^o\}_{i=1}^T) = \sum_{i=1}^T E_c(\mathbf{p}_i) + \sum_{i=1}^{T-1} E_c(\Delta \mathbf{p}_{ij}), \quad (7)$$

where E_c is the constraint energy from Equation (1). We use PGO for absolute pose refinement from consecutive relative poses (see Section III-B1) and absolute pose refinement from relative poses from the RPR model (see Section III-B2).

1) *PGO for APR*: PGO refines the predicted absolute poses such that the relative transforms between them agree with the relative camera pose between the predicted poses. To achieve this, we use a time-distributed image encoder (GoogLeNet) similar to MapNet [14] while using PGO during inference (see Figure 4). We learn to estimate the 6DoF camera pose from a tuple of images and additionally enforce constraints between pose predictions for each image pair. While the APR encoder minimizes the absolute pose loss per image, MapNet proposes to minimize both the absolute pose loss per image and the relative pose loss between the consecutive image pair as

$$\mathcal{L}_{\text{total}} = \sum_{i=1}^{|D|} h(\mathbf{p}_i, \hat{\mathbf{p}}_i) + \sum_{i=1}^{|D|} h(\mathbf{v}_{ij}, \hat{\mathbf{v}}_{ij}), \quad (8)$$

where the relative pose between predicted absolute poses \mathbf{p}_i and \mathbf{p}_j for image pairs (i, j) is represented by $\mathbf{v}_{ij} = (t_i - t_j, q_i - q_j)$. $h(\cdot)$ is a metric that measures the distance between the actual pose $\hat{\mathbf{p}}$ and the predicted camera pose \mathbf{p} as defined in [70]. During inference, we use PGO to fuse the predicted

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

AUGUST 2023

7

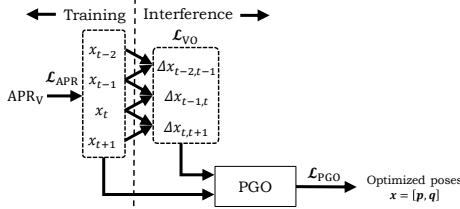


Fig. 4: Overview of the APR_V model with PGO used for absolute pose refinement using the consecutive relative poses between the predicted absolute poses.

absolute poses \mathbf{p}_i and the relative poses (\mathbf{v}_{ij}) between the consecutive image pairs to get smooth and globally consistent absolute poses \mathbf{p}_i^o .

2) APR_V-RPR_I+PGO: MapNet+ [14] relies on visual data to regress the absolute poses (that fail to provide accurate information in challenging conditions) and show how the geometric constraints between pairs of observations can be included as an additional loss term (i.e., VO from pairs of images, from GPS readings, or from IMU readings). Suppose we have IMU sequences of the same scene as additional data. The relative poses can be regressed using these sequences in order to efficiently update the graph, particularly in challenging lighting conditions. Therefore, our fusion model consists of APR_V and RPR_I models to simultaneously regress absolute and relative poses for S pairs of images and IMU sequences sampled with a gap of k timesteps from the dataset D (see Figure 5). The APR_V and RPR_I networks process the images and IMU sequences separately in a time-distributed manner. The visual features (\mathbf{a}_V) from APR_V and inertial features (\mathbf{a}_I) from RPR_I of the last layers are fused based on the soft fusion mechanism discussed in Section III-C2. The features from the soft fusion model are forwarded to BiLSTM layers and a pose regressor, one for each absolute and relative pose regression. We use a similar optimization method as proposed in MapNet [14]. The minor difference is that, while MapNet optimizes the prediction of absolute pose \mathbf{p}_i and \mathbf{p}_j for image pairs (i, j) and the relative pose \mathbf{v}_{ij} between (i, j) , our fusion model additionally optimizes the relative pose (Δt_{ij} , Δq_{ij}) regressed by the RPR_I model. So the final loss function for the fusion model is

$$\begin{aligned} \mathcal{L}_{\text{total}} = & \gamma \sum_{i=1}^{|D|} h(\mathbf{p}_i, \hat{\mathbf{p}}_i) + \alpha \sum_{i,j=1, i \neq j}^{|D|} h(\mathbf{v}_{ij}, \hat{\mathbf{v}}_{ij}) + \\ & + \beta \sum_{i,j=1, i \neq j}^{|D|} h(\Delta \mathbf{p}_i, \Delta \hat{\mathbf{p}}_i), \end{aligned} \quad (9)$$

the sum of losses for the predicted absolute poses \mathbf{p}_i , the relatives poses between the predicted absolute poses \mathbf{v}_{ij} , and the predicted relative poses $\Delta \mathbf{p}_i$ weighted by the hyperparameters α , β and γ . We utilize Optuna to search for optimal hyperparameters. During inference, we use PGO to fuse the predicted absolute poses \mathbf{p}_i and the predicted relative pose $\Delta \mathbf{p}_{ij}$ from the fusion network to get smooth and globally consistent absolute poses \mathbf{p}_i^o .

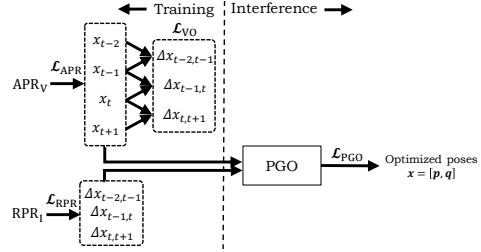


Fig. 5: Overview of the APR_V-RPR_I fusion model with PGO used for absolute pose refinement using the predicted relative poses from the RPR_I model.

C. Attention-based Fusion Methods

Visual and inertial features offer different strengths to pose regression. Hence, our objective is to extract meaningful information from the camera and IMU sensors and to obtain a precise estimate of the absolute poses through the use of a combined feature representation. Inspired by widely applied attention mechanisms [79], [136], [146], we re-weight each feature by conditioning on both visual and inertial features. The selection process of attention-based fusion is conditioned on the measurement reliability and the dynamics of both sensors, which learns to keep the most relevant feature representations while discarding useless or noisy information. For decision level fusion of APR_V and RPR_I features, we use layer concatenation (see Section III-C1), and soft fusion [25] (see Section III-C2). We use the architecture introduced by [65] to fuse the APR_V and RPR_I features at the intermediate levels (see Section III-C3).

1) *Late Fusion (Layer Concatenation)*: We visualize late fusion in Figure 6 that combines the high-level features produced by the individual sources to extract meaningful information for future pose regression tasks. Late fusion is possible when the features have the same number of units and dimensionality. The structure of the late fusion based on concatenation is shown in Figure 7. The fusion network consists of the baseline models APR_V and RPR_I (see Section III-A), that process the image and IMU data separately. During fusion, we concatenate the 1D visual features \mathbf{a}_V from APR_V and 1D inertial features \mathbf{a}_I from RPR_I. \mathbf{a}_V and \mathbf{a}_I are of size 128. In order to model the temporal dependencies between the combined features, we add a two-layer BiLSTM. After the recurrent network, an FC layer is utilized to regress the absolute and relative poses in an MTL setup.

2) *Late Fusion (Soft Fusion)*: The structure of the soft fusion module is shown in Figure 8. We combine the high-level features \mathbf{a}_V and \mathbf{a}_I produced by the APR_V and RPR_I models. The features are fused based on the selective sensor fusion (SSF) approach introduced by Chen et al. [25] that contains an attention mechanism. A pair of continuous masks, $\mathcal{S}_{\text{APR}_V}$ and $\mathcal{S}_{\text{RPR}_I}$, are introduced by

$$\begin{aligned} \mathcal{S}_{\text{APR}_V} &= \text{Sigmoid}_V([\mathbf{a}_V; \mathbf{a}_I]) \\ \mathcal{S}_{\text{RPR}_I} &= \text{Sigmoid}_I([\mathbf{a}_V; \mathbf{a}_I]) \end{aligned} \quad (10)$$

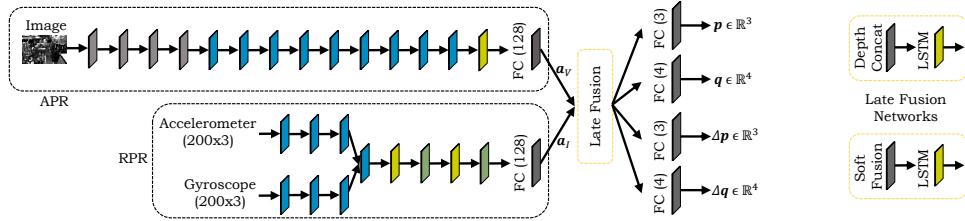


Fig. 6: Overview of the fusion of the visual feature \mathbf{a}_V of the APR_V model and the inertial feature \mathbf{a}_I of the RPR_I model with layer concatenation. Four FC layers regress the absolute pose $[\mathbf{p}, \mathbf{q}]$ and the relative pose $[\Delta\mathbf{p}, \Delta\mathbf{q}]$.

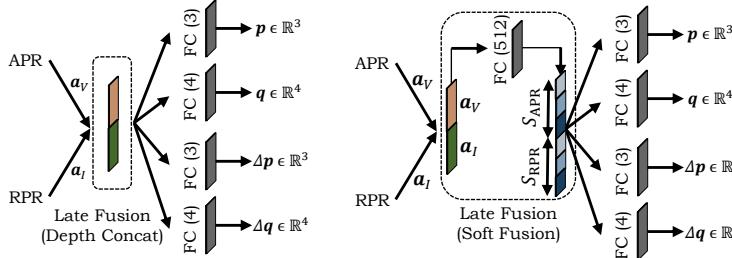


Fig. 7: Overview of the fusion of the visual feature \mathbf{a}_V of the APR_V model and the inertial features \mathbf{a}_I of the RPR_I model using concatenation. Four FC layers regress the absolute pose $[\mathbf{p}, \mathbf{q}]$ and the relative pose $[\Delta\mathbf{p}, \Delta\mathbf{q}]$.

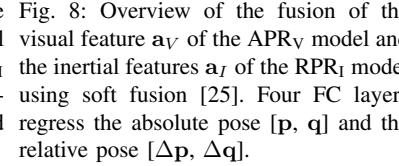


Fig. 8: Overview of the fusion of the visual feature \mathbf{a}_V of the APR_V model and the inertial features \mathbf{a}_I of the RPR_I model using soft fusion [25]. Four FC layers regress the absolute pose $[\mathbf{p}, \mathbf{q}]$ and the relative pose $[\Delta\mathbf{p}, \Delta\mathbf{q}]$.

to perform soft fusion. S_{APR_V} and S_{RPR_I} are the masks applied (element-wise product \odot) to the features \mathbf{a}_V and \mathbf{a}_I learned by the CNNs by conditioning on both features by

$$g_{\text{soft}}(\mathbf{a}_V, \mathbf{a}_I) = [\mathbf{a}_V \odot S_{\text{APR}_V}; \mathbf{a}_I \odot S_{\text{RPR}_I}]. \quad (11)$$

The sigmoid function finally re-weights each feature vector and preserves the order of coefficient values in the range $[0, 1]$ to produce the new re-weighted vectors. After the soft-fusion, the features are forwarded to the recurrent network and finally to FC layers to perform the pose regression.

3) *Intermediate Fusion (MMTM)*: As the APR_V and RPR_I models have unaligned spatial dimensions, they cannot directly be fused using the commonly used techniques like element-wise summation [56], weighted average [105], or more sophisticated methods like attention mechanisms [25] that assume identical spatial dimensions of different streams. Inspired by the squeeze-and-excitation (SE) [57] module for unimodal CNNs, Joze et al. [65] proposed the multimodal transfer module (MMTM) that allows the fusion of modalities with different spatial dimensions. Importantly, the squeeze operation squeezes the spatial information into channel descriptors via a global average pooling operation over spatial dimensions of input features that enables fusion of modalities of arbitrarily feature dimension. The excitation operation generates the excitation signals using a simple gating mechanism as a sigmoid function, which allows the suppression or excitation of different filters in each stream. While MMTM was applied to RGB+depth, RGB+wave, and RGB+pose (both modalities correspond to each other without prerequisites), the fusion of APR and RPR with MMTM proved to be more

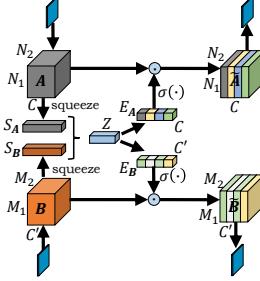
challenging, as RPR requires knowledge of its global pose. In our approach, this issue was addressed by utilizing a time-distributed PoseNet, which provided absolute poses from three consecutive images, see [109]. The structure of MMTM [65] is shown in Figure 9. The matrices $\mathbf{A} \in \mathbb{R}^{N_1 \times \dots \times N_K \times C}$ and $\mathbf{B} \in \mathbb{R}^{M_1 \times \dots \times M_L \times C'}$ represent the features at any given level of the APR_V and RPR_I models that are the inputs to the MMTM module. N_i and M_i represent the spatial dimensions, and C and C' represent the number of channels of APR_V and RPR_I , respectively. MMTM learns the global multimodal embedding to re-calibrate the inputs \mathbf{A} and \mathbf{B} using the SE operation on the input tensors \mathbf{A} and \mathbf{B} . The squeeze operation enables fusion between the modalities S_A and S_B that have arbitrary spatial dimensions.

$$\begin{aligned} S_{\mathbf{A}}(c) &= \frac{1}{\prod_{i=1}^K N_i} \sum_{n_1, \dots, n_K} \mathbf{A}(n_1, \dots, n_K, c) \\ S_{\mathbf{B}}(c) &= \frac{1}{\prod_{i=1}^L M_i} \sum_{m_1, \dots, m_L} \mathbf{B}(m_1, \dots, m_L, c) \end{aligned} \quad (12)$$

that are further mapped into a joint representation Z using concatenation and FC layers. Excitation signals, $E_{\mathbf{A}} \in \mathbb{R}^C$ and $E_{\mathbf{B}} \in \mathbb{R}^{C'}$ are generated using Z , which are used to re-calibrate the input features, \mathbf{A} and \mathbf{B} , by a simple gating mechanism

$$\begin{aligned} \tilde{\mathbf{A}} &= 2 \times \sigma(E_{\mathbf{A}}) \odot \mathbf{A} \\ \tilde{\mathbf{B}} &= 2 \times \sigma(E_{\mathbf{B}}) \odot \mathbf{B}, \end{aligned} \quad (13)$$

where $\sigma(\cdot)$ is a sigmoid function and \odot is a channel-wise product operation. We use MMTM to fuse the features of the



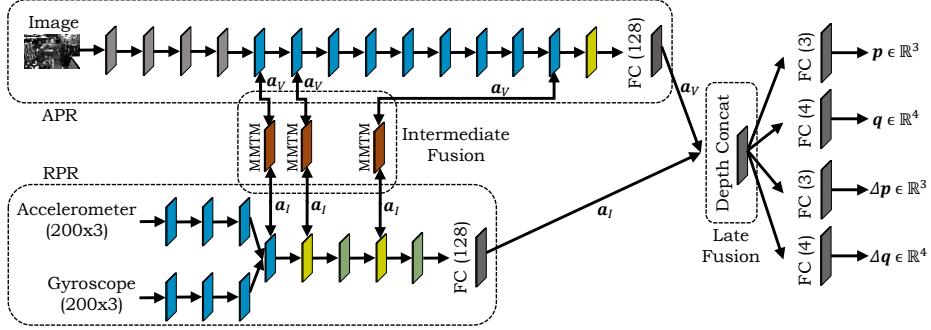


Fig. 10: Overview of the fusion of intermediate visual features a_V of the APR_V model and intermediate inertial features a_I of the RPR_I model with three MMTMs [65] (see Figure 9) for latent layer representations. After late fusion with layer concatenation, four FC layers regress the absolute pose $[p, q]$ and the relative pose $[\Delta p, \Delta q]$.

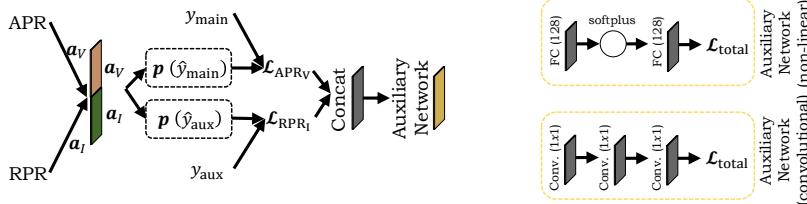


Fig. 11: Left: Overview of the structure of APR_V and RPR_I fusion with the auxiliary learning model [106]. Right: After layer concatenation a convolutional or non-linear auxiliary network is employed that learns the combinations of losses.

APR_V and RPR_I models as proposed in Figure 10. Learning the joint representation using MMTM allows the APR_V model to re-calibrate the features of RPR_I when the IMU sensor data is noisy, or vice versa when the images are blurred, textureless, or are low light. Based on the experiments conducted in [65], the best performance is achieved when the output of half of the last modules of two uni-modal streams are fused by MMTM modules. We insert the first MMTM module at layer 12 of the APR_V model and at layer 5 of the RPR_I model. Then add the second MMTM at layer 14 of APR_V and layer 7 of RPR_I , and the third MMTM at layer 15 of APR_V and at layer 9 of RPR_I (see Figure 10). Finally, similar to the late fusion, the combined features at the end are concatenated and forwarded to the FC layers to regress the poses.

D. Auxiliary Learning

In the low data regime, where the main task overfits and generalizes poorly to unseen data, learning auxiliary tasks have proven to benefit the learning process [63]. In this work, we use the auxiliary learning framework AuxiLearn [106] to optimize the learning of the main task (APR_V) while using RPR_I as auxiliary task. The structure of the AuxiLearn network consists of the main network and an auxiliary network, as shown in Figure 11. The main network is an APR_V - RPR_I soft fusion network (see Section III-C2) that regresses the absolute and relative poses separately. The main network minimizes the losses on the main task \mathcal{L}_{APR_V} and the auxiliary task \mathcal{L}_{RPR_I} . The auxiliary network finally operates on the concatenated vector of losses from the main task. We employ two kinds of auxiliary

networks (see Figure 11): The *convolutional network* variant of the auxiliary network consists of stacked 1D convolutional layers that models the spatial relation among losses, whereas the *non-linear* variant consists of stacked FC layers along with a softplus activation function that captures complex interactions between tasks and learns the non-linear combination of losses. To train the auxiliary learning framework, we use the training set $(\mathbf{x}^t, \mathbf{y}^t)$, and the distinct, independent set $(\mathbf{x}^a, \mathbf{y}^a)$ that represents the auxiliary set. The weights of the main network \mathbf{W} are optimized on the training set $(\mathbf{x}^t, \mathbf{y}^t)$ to minimize the total loss

$$\mathcal{L}_{total} = \mathcal{L}_{APR_V}(\mathbf{x}^t, \mathbf{y}^t; \mathbf{W}) + h(\mathbf{x}^t, \mathbf{y}^t; \mathbf{W}; \phi), \quad (14)$$

where \mathcal{L}_{APR_V} is the loss of the main task, and h is the overall auxiliary loss controlled by ϕ . The loss on the auxiliary set is defined as $\mathcal{L}_A = \mathcal{L}_{APR_V}(\mathbf{x}^a, \mathbf{y}^a; \mathbf{W})$ as we are interested in the generalization performance of the main task. Since there is an indirect dependence of the \mathcal{L}_A on the auxiliary parameters ϕ , we compute the *bi-level optimization* [106] over the main network's parameters \mathbf{W} . In practice, we simultaneously train both, \mathbf{W} and ϕ , by altering between optimizing \mathbf{W} on \mathcal{L}_{total} and ϕ on \mathcal{L}_A .

E. Bayesian Learning

Understanding the uncertainty of a model is a crucial part of many ML systems. Neural networks learn the powerful representations that can map high-dimensional data to an array of features [72], [132]. However, the mappings are often assumed to be accurate, which is not always the case. In order

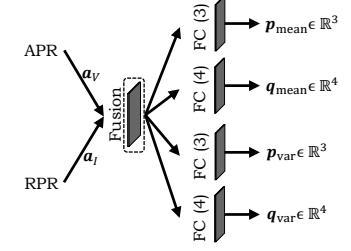


Fig. 12: Overview of the structure of the BNN [71].

to understand the confidence of the models' predictions, we use the Bayesian neural network (BNN) [71] technique, which offers to understand the ML model's uncertainty. Kendall et al. [71] introduced two main kinds of uncertainty: *Aleatoric uncertainty* that captures noise inherent in the training data. *Epistemic uncertainty*, also known as model uncertainty, accounts for uncertainty in the model parameters; this uncertainty can be explained away given enough data. In this work, we model the aleatoric uncertainty by modifying the APR_V-RPR_I fusion architecture to predict both, the mean pose values \mathbf{p}_{mean} and the corresponding variance σ_p^2 (see Figure 12). This modification induces a new kind of minimization objective based on the aleatoric uncertainty as

$$\mathcal{L}_{\text{BNN}} = \frac{\|\mathbf{p} - \hat{\mathbf{p}}\|^2}{2\sigma_p^2} + \frac{1}{2} \log \sigma_p^2, \quad (15)$$

where \mathbf{p} and $\hat{\mathbf{p}}$ are the predicted and ground truth absolute poses and σ_p^2 are the predicted variances. We do not need the uncertainty labels to learn the uncertainty, rather, we only need to supervise the learning of the pose regression by learning the variances implicitly from the loss function. If the model is uncertain in pose prediction (first term of Equation (15)), the model learns to attenuate the total loss \mathcal{L}_{BNN} by increasing the uncertainty σ_p^2 . The second regularization term of Equation (15), however, prevents the network from predicting infinite uncertainty, and thus can be thought of as "learned loss attenuation". In practice, we train the network to predict the log variance $s_p := \log \sigma_p^2$ with the loss

$$\mathcal{L}_{\text{BNN}} = \frac{1}{2} \exp(-s_p) \|\mathbf{p} - \hat{\mathbf{p}}\|^2 + \frac{1}{2} s_p. \quad (16)$$

To regress s_p is more stable than to regress σ_p^2 that avoids a potential division by zero and dampens the effect of log-functions.

F. Deep Multimodal Fusion for Relative Pose Regression

In this section, we propose techniques for VI odometry by fusing the relative pose regression models RPR_V and RPR_I as discussed in Section III-A. RPR_V extracts the latent representation from two consecutive monocular images, and RPR_I extracts temporal information from the inertial data. Both RPR_V and RPR_I models are supervised to regress the relative change in pose. In order to combine the high-level features produced by the two encoders from raw data sequences, we perform the late fusion (Section III-C2) and intermediate fusion (Section III-C3) of RPR_V and RPR_I to regress the relative poses. Furthermore, we use the BNN [71] (Section III-E) to model the aleatoric uncertainty in the relative pose estimation. It is not possible to perform PGO for RPR_V and RPR_I fusion since PGO aims to optimize the consecutive absolute poses. Similarly, we cannot use the auxiliary learning framework [106] as it involves learning two related tasks namely, the main task of interest, while using another auxiliary task to aid the learning of the main task.

Late Fusion. To perform late fusion, we use a similar architecture as in Section III-C2. The fusion model takes high-level visual features \mathbf{a}_V and inertial features \mathbf{a}_I from image

and IMU encoders, respectively. Contrarily, the visual features input \mathbf{a}_V for the fusion model is obtained from the output of layer Conv6, rather than being derived from the last layer of APR_V. The inertial features \mathbf{a}_I from the RPR_I model remain the same. We perform soft fusion [25] by generating a pair of continuous masks, $\mathcal{S}_{\text{RPR}_V}$ and $\mathcal{S}_{\text{RPR}_I}$ from the visual features \mathbf{a}_V and the inertial features \mathbf{a}_I (see Equation (10)). Finally, the output of the fusion model g_{soft} is propagated further to the FC layers to perform pose regression.

Intermediate Fusion. We utilize MMTM [65] as discussed in Section III-C3 to perform the intermediate fusion of the RPR_V and RPR_I models. We insert the first MMTM at layer 7 of RPR_V and layer 5 of RPR_I, the second MMTM at layer 8 of RPR_V and layer 7 of RPR_I, and the third MMTM at layer 9 of RPR_V and layer 9 of RPR_I. Finally, similar to the late fusion, the features from the last layers of RPR_V and RPR_I are concatenated and forwarded to the FC layers to regress the relative pose.

Bayesian Learning. To model the aleatoric uncertainty inherent in the relative pose regression of the RPR_V-RPR_I fusion model, we follow a similar method as discussed in Section III-E. First, we modify the RPR_V-RPR_I fusion architecture based on late fusion to predict both the mean relative poses and their corresponding variances. Finally, we adapt Equation (15) based on the aleatoric uncertainty to minimize the loss in the pose regression as

$$\mathcal{L}_{\text{BNN}} = \frac{1}{2} \exp(-s_{\Delta p}) \|\Delta \mathbf{p} - \Delta \hat{\mathbf{p}}\|^2 + \frac{1}{2} s_{\Delta p}. \quad (17)$$

where $s_{\Delta p} := \log \sigma_{\Delta p}^2$. $\Delta \mathbf{p}$ and $\Delta \hat{\mathbf{p}}$ are the predicted and ground truth relative poses, and $\sigma_{\Delta p}^2$ is the predicted variance.

IV. DATASETS

We give details about the EuRoC MAV and PennCOSYVIO datasets in Section IV-A, respectively in Section IV-B. We propose our novel IndustryVI dataset in Section IV-C.

A. The EuRoC MAV Dataset

The EuRoC MAV [18] dataset was collected on-board an MAV and was recorded in an industrial machine hall (MH) environment and in an indoor Vicon (V) room. The dataset contains synchronized images (from a front-down looking stereo camera), IMU measurements, and ground truth poses from a Leica Nova MS50 laser tracking system and a motion capture system. Exemplary images are given in Figure 13. 11 datasets range from slow flights under good visual conditions (MH-01, MH-02, V1-01, V2-01) to dynamic flights with poor illumination (MH-04, MH-05) and motion blur (MH-03, V1-02, V1-03, V2-02, V2-03). This presents a difficulty in terms of generalizing the dataset. The size of the Vicon room is small-scale ($8m \times 8.4m \times 4m$). Many (SLAM) methods are evaluated on this dataset as it contains different motion dynamics, but the dataset is not useful for many applications (i.e., robotics or handheld devices) as it is recorded on an MAV. The dataset contains 14,566 image and 145,660 IMU training samples and 12,481 image and 124,810 IMU test samples. We train on MH-01, MH-03, and MH-04 and test on MH-02 and MH-05 for APR techniques, and cross-validate all sequences for RPR techniques.



Fig. 13: Exemplary images of the EuRoC MAV dataset [18] of the machine hall (1-3) and the Vicon room (4).



Fig. 14: Images of the PennCOSYVIO dataset [112].



Fig. 15: Environment setup of the IndustryVI dataset with high absorber walls, randomly placed objects and warehouse racks.



Fig. 16: Exemplary images with texture-less surfaces and variations in illuminations of the IndustryVI dataset.

B. The PennCOSYVIO Dataset

The PennCOSYVIO [112] dataset was recorded with a handheld device at the University of Pennsylvania's Singh center. The device contains a stereo camera, an IMU, two Tango devices, and three GoPro cameras. An 150m long path crosses from outdoors to indoors. Four sequences include rapid rotations, changes in lighting, repetitive structures such as large glass surfaces, and different textures (see Figure 14). The sequences AF and AS are for training and BF and BS are for testing. AS and BS are recorded at slow pace, and AF and BF at fast pace. We train and test on each pace separately. The dataset contains 5,035 image and 50,318 IMU training samples and 5,369 image and 53,670 IMU test samples.

C. The IndustryVI Dataset

Given that the EuRoC MAV dataset was captured in an industrial environment but its dynamics of the MAV are distinct from the dynamics of many robotic or handheld systems, and the PennCOSYVIO dataset was recorded in an environment that is different to industrial circumstances, we have recorded a novel dataset in a large-scale industrial environment. The environment is similar to [88], [109]. The visual-only Warehouse [88] dataset (Industry scenario #1) was captured utilizing a (robot-like) positioning system, and its eight diverse testing scenarios offer opportunities for assessments

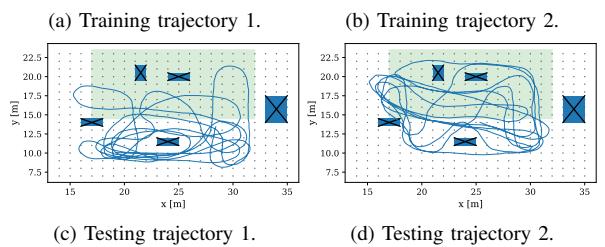
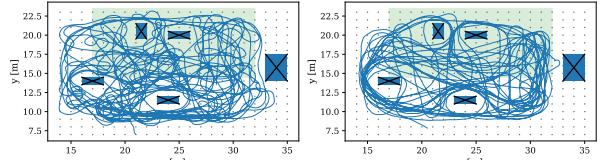


Fig. 17: Training and test trajectories of the IndustryVI dataset.

of generalizability, volatility, and scale transition. The visual-only Industry scenario #2 [109] dataset allows an evaluation for different camera angles. The visual-only Industry scenario #3 [109] dataset was captured on a forklift truck to evaluate for high motion dynamics. As these datasets cover only image data, we record and publish the IndustryVI (scenario #4) dataset. Our environment covers an area of $1,320m^2$, and contains five large black absorber walls and several smaller objects (see Figure 15). We built a handheld device with an Orbbee3D camera with a RGB image resolution of 640×480 pixels and a recording frequency of 23 Hz with an integrated IMU at 140 Hz. Exemplary images are shown in Figure 16. We use a high-precision ($< 1mm$) motion capture system for measuring reference poses at 140 Hz. We let two persons randomly walk in the environment. Trajectories are shown in Figure 17, where training (a) and testing (c) trajectory 1 is from person 1, and training (b) and testing (d) trajectory 2 is from person 2. This results in a total of 55,973 image and 340,620 IMU training samples and 13,990 image and 85,120 IMU test samples. We cross-validate the training and test trajectories. This allows an evaluation between different motion dynamics in a large-scale environment with texture-less ambiguous elements.

V. EXPERIMENTAL RESULTS

Hardware & Training Setup. For all experiments, we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the Adam optimizer with a learning rate of 10^{-4} . We run each experiment for 1,000 epochs with a batch size of 50 and report results for the best epoch.

Evaluation Metrics. For the evaluation of the APR, we report the median absolute position $e_{med,p}$ in m and the median absolute orientation $e_{med,q}$ in $^\circ$, and the median relative position $\Delta e_{med,p}$ in Δm and the median relative orientation $\Delta e_{med,q}$ in Δ° . As the global consistency of the estimated trajectory is an important quantity and to compare our relative pose prediction with state-of-the-art techniques, we additionally report the absolute trajectory error (ATE) by aligning the estimated

TABLE III: Median absolute $\mathbf{p}(m/\circ)$ and relative $\Delta\mathbf{p}(\Delta m/\Delta\circ)$ pose estimation results for the EuRoC MAV [18] dataset.

Method	MH-02-easy				MH-04-difficult			
	$\mathbf{p}(m/\circ)$		$\Delta e_{\text{med},p}$	$\Delta e_{\text{med},q}$	$\mathbf{p}(m/\circ)$		$\Delta e_{\text{med},p}$	$\Delta e_{\text{med},q}$
	$e_{\text{med},p}$	$e_{\text{med},q}$			$e_{\text{med},p}$	$e_{\text{med},q}$		
APR _V : PoseNet [132]	0.9249	3.1718	-	-	0.9405	3.1086	-	-
RPR _I : IMUNet [36]	-	-	0.0276	0.0741	-	-	0.0310	0.1073
MapNet [14]	0.9859	3.1879	-	-	0.9905	3.1898	-	-
MapNet+PGO [14]	0.9435	3.1656	-	-	0.9690	3.1156	-	-
APR _V -RPR _I +PGO [50]	0.6914	3.1050	0.0350	0.4124	0.7211	3.1532	0.0352	0.4111
Late Fusion (concat)	0.9079	3.1232	0.0381	0.6801	0.9777	3.1134	0.0483	0.5832
Late Fusion (concat) + BiLSTM	0.7902	3.1452	0.0299	0.5062	0.8391	3.1164	0.0471	0.5604
Late Fusion (SSF) [25]	0.9739	3.1744	0.0567	0.6850	0.9254	3.1164	0.0453	0.9805
Late Fusion (SSF) [25] + BiLSTM	0.8114	3.1538	0.0296	0.5670	0.7862	3.1277	0.0240	0.5690
MMTM [65] (3 modules)	0.8356	3.1782	0.0194	0.0601	1.1218	3.1207	0.0202	0.0800
AuxiLearn (non-linear) [106]	0.8612	3.0266	0.0371	0.5671	0.7979	3.0996	0.0631	0.5851
AuxiLearn (convolutional) [106]	0.9050	3.1984	0.0371	0.5561	0.8711	3.1047	0.0612	0.5873
BNN [71] + Late Fusion	0.7925	3.1878	-	-	0.8523	3.0285	-	-

TABLE IV: Median absolute $\mathbf{p}(m/\circ)$ and relative $\Delta\mathbf{p}(\Delta m/\Delta\circ)$ pose estimation results for the PennCOSYVIO [112] dataset.

Method	BF				BS			
	$\mathbf{p}(m/\circ)$		$\Delta e_{\text{med},p}$	$\Delta e_{\text{med},q}$	$\mathbf{p}(m/\circ)$		$\Delta e_{\text{med},p}$	$\Delta e_{\text{med},q}$
	$e_{\text{med},p}$	$e_{\text{med},q}$			$e_{\text{med},p}$	$e_{\text{med},q}$		
APR _V : PoseNet [132]	1.8210	3.1129	-	-	1.4125	3.1411	-	-
RPR _I : IMUNet [36]	-	-	0.1091	1.0573	-	-	0.0393	0.5714
MapNet [14]	3.3017	3.1146	-	-	3.2557	3.1317	-	-
MapNet+PGO [14]	3.4130	3.1211	-	-	3.8911	3.1412	-	-
APR _V -RPR _I +PGO [50]	2.5563	3.1016	0.0402	0.7134	2.3142	3.1360	0.0200	0.7099
Late Fusion (concat)	2.2365	3.1028	0.0385	0.8305	2.0696	3.1390	0.1013	0.9348
Late Fusion (concat) + BiLSTM	1.6543	3.0962	0.0281	0.8162	1.7389	3.1309	0.0974	1.0773
Late Fusion (SSF) [25]	1.8693	3.1021	0.0321	0.8213	1.6552	3.1356	0.0863	0.8762
Late Fusion (SSF) [25] + BiLSTM	1.1249	3.1037	0.0180	0.7571	1.2341	3.1287	0.0291	0.8123
MMTM [65] (3 modules)	1.0557	3.1378	0.0328	0.6695	1.1980	3.1008	0.0976	0.9073
AuxiLearn (non-linear) [106]	1.5402	3.0944	0.0410	1.0195	1.3008	3.1397	0.0525	1.0881
AuxiLearn (convolutional) [106]	1.8931	3.1098	0.0451	1.0220	1.8964	3.1401	0.1006	1.1020
BNN [71] + Late Fusion	2.1110	3.1136	-	-	1.6569	3.1450	-	-

trajectory $\mathbf{P}_{1:n}$ and the ground truth trajectory $\mathbf{Q}_{1:n}$ using the method of Horn [55]. The ATE at time step i can be computed as $\mathbf{F}_i := \mathbf{Q}_i^{-1} \mathbf{S} \mathbf{P}_i$ with the rigid-body transformation \mathbf{S} corresponding to the least-squares solution that maps $\mathbf{P}_{1:n}$ onto $\mathbf{Q}_{1:n}$. Next, we compute the root mean squared error over all time steps of the translational components by

$$e_{\text{ATE},p}(\mathbf{F}_{1:n}) := \left(\frac{1}{n} \sum_{i=1}^n \|\text{trans}(\mathbf{F}_i)\|^2 \right)^{\frac{1}{2}}. \quad (18)$$

To compare our RPR results with the results proposed by [25], [36], we use the absolute translational error (ATLE) [36] for the position $\Delta e_{\text{ATLE},p}$ in m .

A. Evaluation of APR_V-RPR_I Fusion Methods

We provide a quantitative evaluation of all APR_V-RPR_I fusion methods for the EuRoC MAV (Table III), the PennCOSYVIO (Table IV), and the IndustryVI (Table V) datasets. For an overview of APR trajectory comparisons, see Figure 22 to Figure 27 in the appendix.

Baseline Results, MapNet, and PGO. We evaluate the results for MapNet [14], PGO, and PGO for APR_V-RPR_I fusion, and compare the results to the baseline methods. The APR_V baseline yields proper results of $0.9249m$ and $0.9405m$ on the small-scale environment of EuRoC MAV, and RPR_I yields small errors of $2.76cm$ and $3.1cm$ (even at fast dynamics of the MAV). This increases for the large-scale area of PennCOSYVIO and IndustryVI. The relative error of RPR_I

(below $3.0cm$) is low for the fast movements of the IndustryVI dataset. While MapNet and MapNet+PGO (see Section III-B1) increase the APR_V baseline results for the EuRoC MAV and PennCOSYVIO datasets and most sequences of the IndustryVI dataset, our implementation of APR_V-RPR_I fusion utilizing PGO (see Section III-B2) yields notably lower results on the EuRoC MAV dataset (e.g., $0.6914m$ on the MH-02 sequence compared to $0.9249m$ of APR_V-only) and the train 2 dataset of IndustryVI. For the PennCOSYVIO dataset, the APR_V-RPR_I fusion utilizing MapNet+PGO cannot outperform PoseNet. This contradicts the results of MapNet and MapNet+PGO on the 7-Scenes [125] dataset proposed in [14], where MapNet+PGO significantly improves the PoseNet results. Given that MapNet and MapNet+PGO are time-distributed networks, it is possible to enhance their performance by increasing the training time steps and incorporating larger skip sizes between consecutive images.

Late Fusion (Concatenation). Next, we evaluate the late fusion of APR_V-RPR_I utilizing concatenation with and without BiLSTM layers (see Section III-C1). For the EuRoC MAV dataset, the concatenation improves the APR_V-only model for the MH-02 sequence, but decreases for the MH-04 sequence. The concatenation with BiLSTM layers can notably reduce the absolute pose results, but cannot outperform the APR_V-RPR_I+PGO [50] fusion, while the relative position results marginally decrease. For the PennCOSYVIO dataset, the late fusion decreases the model performance for the BF and BS

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

AUGUST 2023

13

TABLE V: Median absolute $p(m/\circ)$ and relative $\Delta p(\Delta m/\Delta\circ)$ pose estimation results for the IndustryVI dataset.

Method	Train 1, Test 1				Train 1, Test 2				Train 2, Test 1				Train 2, Test 2			
	$p(m/\circ)$	$\Delta p(\Delta m/\Delta\circ)$	$e_{med,p}$	$e_{med,q}$	$p(m/\circ)$	$\Delta p(\Delta m/\Delta\circ)$	$e_{med,p}$	$e_{med,q}$	$p(m/\circ)$	$\Delta p(\Delta m/\Delta\circ)$	$e_{med,p}$	$e_{med,q}$	$p(m/\circ)$	$\Delta p(\Delta m/\Delta\circ)$	$e_{med,p}$	$e_{med,q}$
APRV: PoseNet [132]	1.8231	9.6742	-	-	1.6429	7.8564	-	-	1.9345	6.2314	-	-	1.6438	7.3452	-	-
RPR _I : IMUNet [36]	-	-	0.0300	0.8867	-	-	0.0295	0.7743	-	-	0.0278	0.6134	-	-	0.0265	0.9641
MapNet [14]	1.9012	10.120	-	-	1.8990	8.1803	-	-	1.9019	6.9678	-	-	1.6891	8.4012	-	-
MapNet+PGO [14]	1.8865	9.9905	-	-	1.8126	7.9078	-	-	1.8991	6.9120	-	-	1.6694	8.1021	-	-
APRV-RPR _I +PGO [50]	1.8681	9.8901	0.0481	0.9841	1.7106	7.1193	0.0458	0.7925	1.8379	6.2731	0.0419	0.9251	1.6234	7.8351	0.0413	0.9811
Late Fusion (concat)	1.8945	9.8976	0.0342	0.9995	1.9016	8.2087	0.0335	0.8121	1.8154	7.7210	0.0400	1.1301	1.5342	8.6910	0.2107	1.2004
+ BiLSTM	1.6014	5.1032	0.0301	0.9012	1.6521	5.8724	0.0271	0.8610	1.6231	6.0013	0.0242	0.9221	1.3092	6.0123	0.0254	0.7449
Late Fusion (SSF) [25]	1.7613	9.5848	0.0315	0.9491	1.8616	7.1069	0.0321	0.7867	1.7823	7.3210	0.0381	<u>1.2031</u>	1.5005	8.2451	0.1097	1.3164
+ BiLSTM	1.5875	5.2016	0.0214	0.9823	1.5216	5.7630	0.0260	0.7967	1.6823	5.4601	0.0278	0.9231	1.3215	5.0291	0.0257	0.9164
MMTM [65] (3 modules)	1.6550	5.2045	0.0374	0.8179	1.5775	5.8832	0.0443	1.0797	1.7836	5.6034	0.0378	0.9709	1.4840	5.2451	0.0401	0.8726
AuxiLearn (non-linear) [106]	1.6845	6.2312	0.0310	0.9931	1.6140	5.8834	0.0312	0.8127	1.6941	5.8848	0.0305	1.0867	1.3341	5.0221	0.0213	0.8934
AuxiLearn (convol.) [106]	1.8180	9.7841	0.0332	0.9836	1.9011	8.3024	0.0367	0.8651	1.7983	6.9812	0.0361	1.1331	1.5670	7.8938	0.2207	1.1956
BNN [71] + Late Fusion	1.7956	9.7547	-	-	1.7691	8.1268	-	-	1.8251	<u>7.3289</u>	-	-	1.5476	7.8751	-	-

TABLE VI: Intermediate APR_V-RPR_I fusion using MMTM [65] with separate model optimization for the EuRoC MAV [18] dataset. While “F” indicates the layer with MMTM fusion (see Section III-C3), “-” indicates no fusion. For a visualization, see Figure 37 in the appendix. **Bold** are best results. Underline are second best results.

MMTM	MH-02-easy						MH-04-difficult									
	$p(m/\circ)$	$\Delta p(\Delta m/\Delta\circ)$	$e_{med,p}$	e_{ATE}	e_{ATLE}	$e_{med,q}$	$\Delta e_{med,p}$	e_{ATE}	e_{ATLE}	$e_{med,q}$	$\Delta e_{med,p}$	e_{ATE}	e_{ATLE}	$e_{med,q}$	$\Delta p(\Delta m/\Delta\circ)$	
(F - -)	0.8399	3.2603	0.3111	3.1980	0.0205	1.7572	0.0691	0.0661	1.1084	3.9618	0.4497	<u>3.1156</u>	0.0249	3.7785	0.0661	0.0818
(- F -)	0.9512	3.2789	0.3114	3.1940	0.0219	2.0637	0.0705	0.0645	1.1098	4.3766	0.4439	3.1937	0.0253	<u>3.2841</u>	0.0649	0.0751
(- - F)	0.8743	2.5593	0.5797	<u>3.1795</u>	0.0207	2.0813	0.0693	0.0708	<u>1.0619</u>	3.8419	0.7398	3.1047	0.0244	3.6690	0.0658	0.0686
(F F -)	0.9199	3.3279	0.2792	3.1689	0.0213	1.3248	0.0696	0.0661	1.0592	4.1632	0.4205	3.1400	0.0250	3.4385	0.0661	0.0782
(F - F)	0.9601	3.3041	0.3090	3.1483	0.0204	<u>1.5077</u>	0.0710	0.0648	1.0994	4.3420	0.3548	3.1468	0.0227	4.3420	0.0658	0.0771
(- F F)	0.8609	3.3352	<u>0.2897</u>	3.1997	0.0210	1.8346	0.0699	<u>0.0644</u>	1.0996	3.4948	<u>0.3714</u>	3.1664	0.0243	3.4948	0.0638	0.0707
(F F F)	0.8356	<u>2.9978</u>	0.5182	3.1812	0.0194	1.9925	0.0680	0.0601	1.0997	<u>3.5267</u>	0.7560	3.1607	0.0202	2.9733	<u>0.0642</u>	0.0800

sequences, while adding the BiLSTM layers improves the performance for the BF sequence, and hence, outperform the APR_V-only baseline model. Concatenation with BiLSTM layers proves to be effective for the IndustryVI datasets and outperforms all fusion techniques in terms of performance on the train 1 and test 1, train 2 and test 1, and train 2 and test 2 sequences. This demonstration reveals that the straightforward method of concatenating the high-level features does not prove effective in acquiring a meaningful representation between the APR and RPR tasks. The enhancement in performance resulting from the addition of BiLSTM layers underscores the importance of modeling temporal dependencies in achieving successful outcomes for the pose regression tasks. Overall, the error increases for the IndustryVI dataset when training on person 2 and testing on person 1 (increase of position error) and vice versa (increase of orientation error). A good fusion technique can accommodate this, e.g., here, concatenation with BiLSTM.

Late Fusion (SSF). We evaluate the soft fusion approach of SSF [36] as a late fusion method (see Section III-C2). Similar for the late fusion method with concatenation, the BiLSTM layers are a crucial part for the SSF approach. The adoption of SSF and BiLSTM layers leads to a substantial reduction of the absolute position error compared to the APR_V-only baseline model for all three datasets. For example for the PennCOSYVIO BF dataset, SSF with BiLSTM yields a low absolute position error of $1.1249m$, but can still significantly reduce the relative position error from $10.9cm$ to $1.8cm$ and the relative orientation error from 1.057° to 0.757° . This approach demonstrates superior or comparable results compared to the

fusion technique based on concatenation. This highlights the significance of proper feature selection, rather than merely concatenating high-level features.

MMTM. The intermediate fusion method based on MMTM [65] learns a joint representation between APR_V and RPR_I. We train the fusion architecture with seven different combinations of MMTM modules (for details on the fusion layers, see Section III-C3). For the number of trainable model parameters, see Table VIII in the appendix. It is evident that the number of trainable parameters increases as we increase the number of MMTM modules. Table VI summarizes the results for the seven combinations on the EuRoC MAV dataset. The combinations (- - F), (F F -), (F - F), and (F F F) yield comparable results on the EuRoC MAV dataset, while the best and consistent model performances on all three datasets are achieved using the (F F F) combination of three MMTM modules. Consequently, we select the combination of three MMTM modules for the results presented in Tables III to V. For the EuRoC MAV dataset, MMTM (3 modules) decreases the APR results while yielding the best RPR results on the MH-02 sequence. At the expense of the APR error, the RPR error also decreases for MH-04 against the RPR_I baseline. In addition, MMTM yields the best results for the PennCOSYVIO dataset. Our conclusion is that despite being developed for hand gesture recognition, human activity recognition, and audio-visual speech fusion, MMTM proves to be an effective module for learning a joint representation between networks even in the context of the challenging task of fusing APR and RPR.

Auxiliary Learning. Next, we evaluate the AuxiLearn [106]

TABLE VII: RPR_V - RPR_I results given in Δm and Δ° . **Bold** are best results. Underlined are the best RPR baseline results.

Method	EuRoC MAV [18]								PennCOSYVIO [112]				IndustryVI					
	MH-02		MH-04		V1-03		V2-02		V1-01		BF		BS		Test 1		Test 2	
	$\Delta e_{med,p}$	$\Delta e_{med,q}$																
RPR_V : FlowNet [37]	<u>0.0155</u>	0.013	0.0223	0.114	0.0289	0.359	0.0277	0.360	0.0171	0.215	0.0256	0.336	0.0322	0.505	0.0261	0.801	0.0254	0.864
RPR_I : IMUNet [36]	0.0222	0.069	0.0261	0.084	0.0306	0.145	0.0320	0.140	0.0212	0.082	0.1091	0.107	0.0393	0.571	0.0295	0.810	0.0290	0.861
Late Fusion (concat) + BiLSTM	0.0161	0.103	0.0247	0.099	0.0284	0.211	0.0286	0.206	0.0171	0.121	0.0257	0.348	0.0384	0.479	0.0278	0.829	0.0255	0.865
Late Fusion (SSF) [25] + BiLSTM	0.0166	0.079	0.0234	0.088	0.0276	0.200	0.0273	0.215	0.0160	0.116	0.0261	0.343	0.0371	0.498	0.0294	0.860	0.0310	0.886
MMTM [65]	0.0121	0.073	0.0181	0.083	0.0268	0.185	0.0222	0.179	0.0138	0.107	0.0255	0.458	0.0367	0.697	0.0256	0.758	0.0248	0.860

framework (see Section III-D), i.e., the non-linear and convolutional variants. We use the SSF architecture as the main network to optimize the main APR_V task. For all datasets, the non-linear variant yields better results than the convolutional variant. Therefore, it is crucial to model the intricate connections between the APR_V and RPR_I loss functions utilizing a non-linear layer instead of the spatial relationship, which is modeled through the convolutional layer of the auxiliary network. The efficacy of the AuxiLearn model in comparison to the baseline models is contingent upon the sequences. While for MH-02, MH-04, and BS, the APR results increase, the RPR results decrease. The trend is comparable for the IndustryVI dataset. It can be deduced that the RPR_I task serves as a suitable auxiliary task for enhancing the main APR_V task, albeit at the cost of a decreased performance on the RPR_I task. Conversely, the main task does not have a positive impact on the auxiliary task. Therefore, AuxiLearn may prove beneficial for specific self-localization applications that place a significant emphasis on the absolute pose.

Bayesian Neural Networks. The BNN [71] models the aleatoric uncertainty for the APR_V task (see Section III-E). We train the fusion model utilizing the modified loss function in Equation (15). The performance of the BNN is superior to the baseline APR_V model for the EuRoC MAV dataset, evidenced by a reduction in error from $0.9249m$ to $0.7925m$ and from $0.9405m$ to $0.8523m$. Additionally, the BNN demonstrated improved performance for the majority of evaluation sequences of the IndustryVI dataset. However, its performance deteriorates for the PennCOSYVIO dataset. Interestingly, the predicted trajectories are unique and smoother for both the EuRoC MAV dataset (see Figure 22f and Figure 23f in the appendix) and for the IndustryVI dataset. This indicates that the BNN has learned to reduce the variance of the mean prediction values. For the PennCOSYVIO dataset, we observe that the prediction are worse inside the building (as seen in the upper-right part of the Figure 24f and Figure 25f), particularly in regions where the images feature repetitive patterns of extensive glass walls. This phenomenon is also substantiated by the high levels of aleatoric uncertainty present in these areas (see Figure 39). As a result, Bayesian learning may serve as a tool for interpreting complex images, for example, images with difficult illuminations (see Figure 40) or reflective pattern (see Figure 41) can be detected.

B. Evaluation of RPR_V - RPR_I Fusion Methods

We provide quantitative results for the RPR_V - RPR_I fusion task on the EuRoC MAV, PennCOSYVIO, and IndustrialVI

datasets in Table VII. For an overview of RPR trajectory comparisons, see Figure 28 to Figure 36 in the appendix. As the RPR task is independent of the scene geometry, we utilize all training sequences from both scenes (MH and V) of the EuRoC MAV dataset (i.e., MH-01, MH-03, MH-05, V1-02, V2-01, and V2-03) and test on the MH-02, MH-04, V1-01, V1-03, and V2-02 sequences. Hence, the training dataset is large to cover all movement patterns and dynamics of the MAV, while the testing datasets cover the large machine hall and the small living room with different object configurations. First, we evaluate the baseline models FlowNet [37] for the RPR_V task and IMUNet [36] for the RPR_I task. It is noteworthy that the RPR_V model produces superior relative translational results on the EuRoC MAV dataset, while the RPR_I model yields superior relative orientational results. On the PennCOSYVIO dataset, the RPR_V model outperforms the RPR_I model. This shows that the IMU measurements contain a high sensor noise, while the RPR_V model is robust to fast movement changes (of the MAV and of the handheld system). The objective is to merge the advantageous translational predictions of the RPR_V model with the advantageous rotational prediction of the RPR_I model (for the EuRoC MAV dataset), or to selectively choose favorable predictions from the RPR_I model (for the PennCOSYVIO dataset).

Late Fusion (Concatenation). The combination of concatenation with BiLSTM layers can partially enhance the RPR results, particularly for the Vicon datasets within the EuRoC MAV dataset and for the IndustryVI datasets. In contrast, the performance of the late fusion model decreases on the machine hall sequences. Consequently, similar to the APR_V - RPR_I fusion task of the model with concatenation and BiLSTM layers, the performance of the late fusion model is also improved by incorporating BiLSTM layers after fusion to capture temporal dependencies.

Late Fusion (SSF). Soft fusion of RPR_V and RPR_I (see Section III-C2) is only marginally different from the late fusion model with concatenation with a small improvement in the RPR errors. As previously, the performance of the SSF model improves for all datasets by incorporating BiLSTM layers after the fusion. For the EuRoC MAV and IndustrialVI datasets, SSF with BiLSTM layers outperforms both baseline models on all sequences. In the context of PennCOSYVIO, it exhibits superior performance compared to the RPR_I baseline model. However, it is unable to surpass the translational performance of the RPR_V model for both sequences.

MMTM. As for the APR_V - RPR_I fusion, we assess the seven different layer combinations of MMTM for the RPR_V - RPR_I

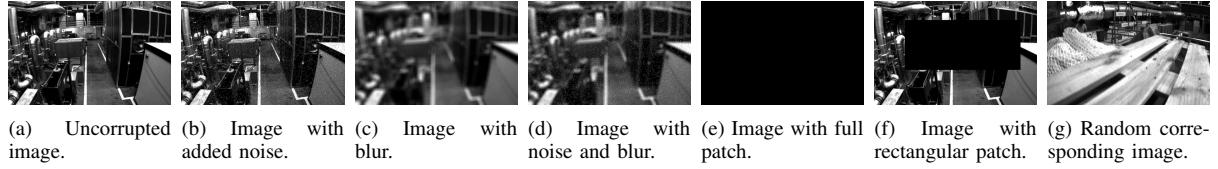


Fig. 18: Image corruption techniques. The original image of the EuRoC MAV [18] dataset is shown in (a).

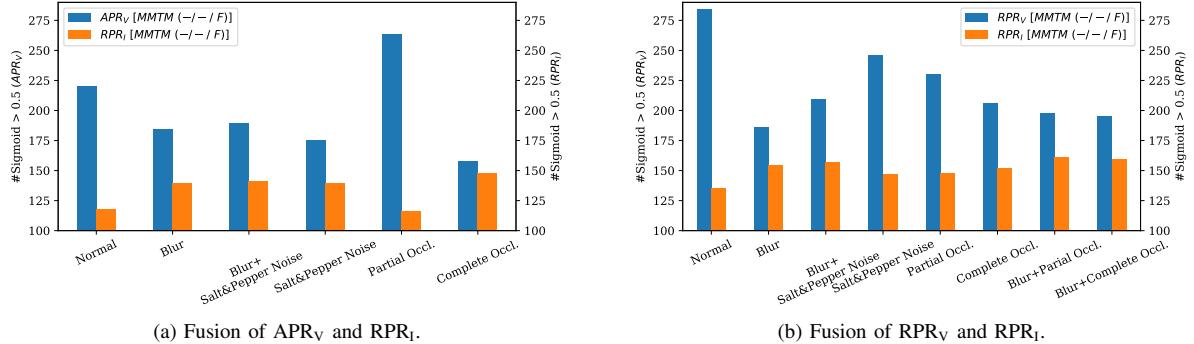


Fig. 19: Number of sigmoid activations higher than 0.5 for APR_V and RPR_V methods (blue) and RPR_I methods (orange) evaluating the third layer for the MMTM module for different image corruption techniques.

fusion (see Table VI). The model featuring three MMTM modules (F F F) for the joint representation exhibits the optimal performance. These results are consistent with the experiments and findings presented by [65]. We note that the fusion with three MMTM modules reduces the translational error of the baseline and fusion techniques on the EuRoC MAV datasets. However, there is not a substantial improvement for the PennCOSYVIO and IndustrialVI datasets.

C. Network Activation Mask Evaluation

In this section, we evaluate the network masks (i.e., the sigmoid activation function outputs) of the MMTM modules in the APR_V - RPR_I and RPR_V - RPR_I fusion tasks. We visualize pairs of the continuous masks E_A and E_B as shown in Figure 9, which depict the feature selection mechanism of the features extracted from the visual and inertial encoders prior to their transmission to the temporal modeling and pose regression. The sigmoid activations ensures that each feature channel is re-weighted within the range of $[0, 1]$ based on its significance at a particular time step. To accomplish this, we corrupt the image input, as depicted in Figure 18a, and apply five distinct corruptions: image with Salt&Pepper noise (Figure 18b), blurred image (Figure 18c), full occlusion (Figure 18e), partial occlusion with a rectangular patch (Figure 18f), and selecting a random corresponding image for RPR_V (Figure 18g). This encompasses a range of real-time scenarios, including fast rotations, occlusions, and low-light conditions. Figure 19 provides the number of sigmoid activations that are higher than 0.5 for the APR_V , RPR_V , and RPR_I regression tasks for all image corruptions. With regards to the APR_V - RPR_I fusion task (see Figure 19a), the number of sigmoid activations decreases for the image

encoder and increases for the inertial encoder for varying image corruptions. This demonstrates that the fusion model progressively relies on the inertial data with an increasing level of image corruption. For the RPR_V - RPR_I fusion model, we corrupt one of the two input images that correspond to each other. We provide the number of sigmoid activations in Figure 19b. The number of activations of the RPR_V model decreases for all image corruptions, while the number of activations increases of the RPR_I model. Particularly, as the level of noise increases (e.g., compare the number of activations for Salt&Pepper noise with the number of activations for the combination of image blur and Salt&Pepper noise), the fusion model becomes more reliant on the inertial encoder. This phenomenon can also be observed when comparing the combination of image blur and partial occlusion or for the combination of image blur and complete occlusion to complete occlusion alone. Upon the comparison of the APR_V - RPR_I fusion with the RPR_V - RPR_I fusion, it can be observed that the RPR_V model contains higher activations compared to the APR_V model. This is due to the increased reliability of the model resulting from the use of two consecutive images, rather than just a single image. Subsequently, we directly visualize the feature selection masks of the MMTM module employed in the APR_V - RPR_I (see Figure 20) and in the RPR_V - RPR_I (see Figure 21) for various image corruptions. The fusion networks learn to assign a greater weight to the inertial features when the images are degraded (as evidenced by the thick green lines of the RPR_I model compared to the APR_V and RPR_V models). This highlights that the networks have learned to place more importance on a complementary sensor in order to perform the regression task in the presence of a challenging image input.

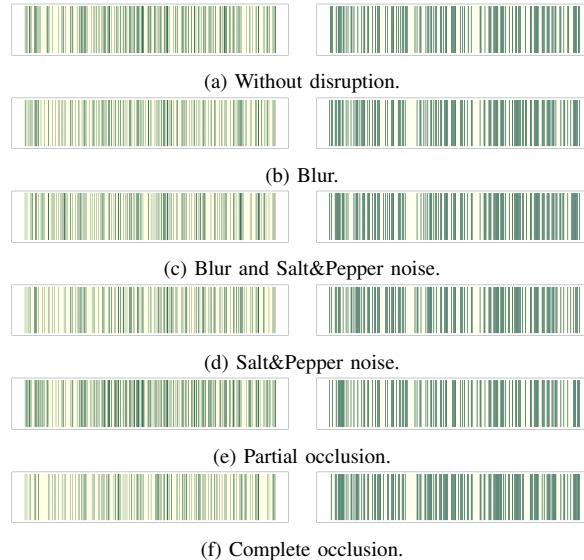


Fig. 20: Activations for the third MMTM module for different image corruption techniques for APR_V (left) and RPR_I (right).

VI. CONCLUSION

We investigated deep multimodal fusion between the visual APR task supported with inertial RPR and between the visual-inertial RPR tasks. As baseline models, we utilized PoseNet for APR_V, IMUNet for RPR_I, and FlowNet for RPR_V. In order to attain globally consistent pose predictions during inference, we analyzed and compared various techniques including MapNet, pose graph optimization, late fusion techniques such as concatenation and selective sensor fusion with BiLSTM layers, intermediate fusion with transfer modules, auxiliary learning, and Bayesian learning. Our assessments on the EuRoC MAV aerial vehicle dataset, the handheld PennCOSYVIO dataset, and our novel large-scale IndustryVI indoor dataset serve as a comprehensive benchmark for the robustness of fusion techniques across various challenging environments and motion dynamics. In conclusion, the results and key findings can be succinctly summarized as follows: (1) The APR_V-RPR_I+PGO approach and the intermediate fusion with the MMTM technique demonstrate superiority over other techniques for the APR_V-RPR_I task on the EuRoC MAV dataset. These methods exhibit an improved capacity for generalization on the dataset with smoother predicted trajectories. (2) Selective sensor fusion and fusion with MMTM exhibit superiority on the PennCOSYVIO and IndustryVI datasets. (3) In addition, the MMTM fusion technique yields the highest performance on the RPR_V-RPR_I task. (4) Fusing three MMTM modules is more advantageous than fusing one or two MMTM modules as it results in a more generalized representation between both modalities. (5) For all the datasets, the non-linear-based auxiliary learning approach enhances the performances of the main task. (6) The estimation of aleatoric uncertainty using Bayesian networks provides valuable insights into the model's robustness against challenging images. (7)

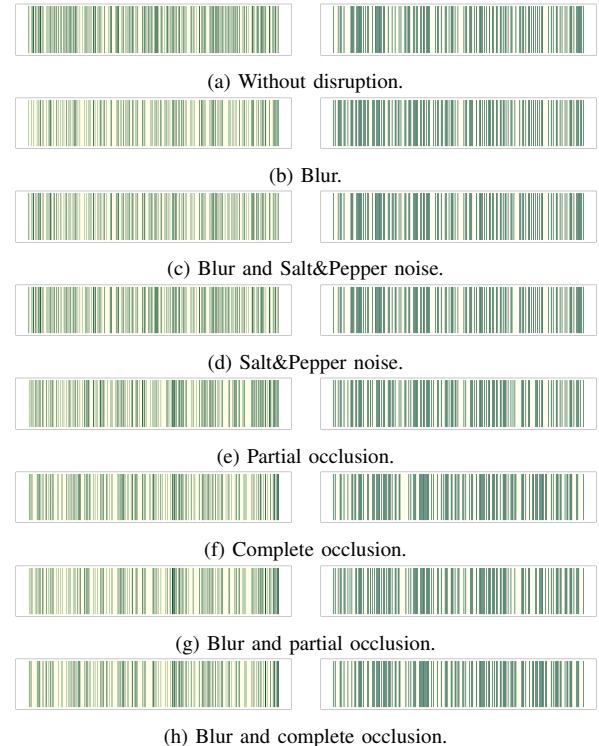


Fig. 21: Activations for the third MMTM module for different image corruption techniques for RPR_V (left) and RPR_I (right).

We examined the network activations by subjecting the image inputs to various disruption techniques. Upon increasing the image corruption, the number of high softmax activations in the visual model increased, whereas the number in the inertial model decreased, indicating higher reliability on the inertial model for challenging images. (8) The IMU bias has a considerable impact on the performance of RPR-only methods, as evidenced by the deviation of the trajectories in the appendix, while the relative pose – even with its inherent noise – can still be effectively utilized to smooth the absolute trajectory.

VII. ACKNOWLEDGMENTS

This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No. 01IS18036A (David Rügamer) and by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications within the framework of “BAYERN DIGITAL II”.

REFERENCES

- [1] Yasin Almalioglu, Mehmet Turan, Alp Eren Sari, Muhamad Risqi U. Saputra, Pedro P. B. de Gusmão, Andrew Markham, and Niki Trigoni. SelfVIO: Self-Supervised Deep Monocular Visual-Inertial Odometry and Depth Estimation. In *arXiv:1911.09968*, November 2019.
- [2] András L. Majdik and Charles Till and Davide Scaramuzza. The Zurich Urban Micro Aerial vehicle Dataset. In *IJRR*, volume 36(3), April 2017.

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

316

AUGUST 2023

17

- [3] Asha Anoosheh, Torsten Sattler, Radu Rimofte, Marc Pollefeys, and Luc Van Gool. Night-to-Day Image Translation for Retrieval-based Localization. In *ICRA*, Montreal, QC, May 2019.
- [4] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous Metric Learning Relocalisation Using Neural Nets. In *ECCV*, pages 751–767, Munich, Germany, September 2018.
- [5] Stéphane Beauregard and Harald Haas. Pedestrian Dead Reckoning: A Basis for Personal Positioning. In *Positioning, Navigation and Communication*, pages 27–35, 2006.
- [6] José-Luis Blanco-Claraco, Francisco Ángel Moreno-Duñas, and Javier González-Jiménez. The Málaga Urban Dataset: High-Rate Stereo and LiDAR in a Realistic Urban Scenario. In *IJRR*, volume 33(2), pages 207–214, October 2014.
- [7] Michael Bloesch, Michael Burri, Sammy Omari, Marco Hutter, and Roland Siegwart. Iterated Extended Kalman Filter Based Visual-Inertial Odometry Using Direct Photometric Feedback. In *IJRR*, volume 36(10), pages 1053–1072, September 2017.
- [8] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust Visual Inertial Odometry Using a Direct EKF-Based Approach. In *IROS*, pages 298–304, Hamburg, Germany, October 2015.
- [9] Matt Bower, Cathie Howe, Nerida McCredie, Austin Robinson, and David Grover. Augmented Reality in Education — Cases, Places, and Potentials. In *ICEM*, Singapore, Singapore, October 2002.
- [10] Eric Brachmann, Martin Humenberger, Carsten Rother, and Torsten Sattler. On the Limits of Pseudo Ground Truth in Visual Camera Relocalisation. In *CVPR*, pages 6218–6228, September 2021.
- [11] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel Stefan Gumhold, and Carsten Rother. DSAC — Differentiable RANSAC for Camera Localization. In *CVPR*, pages 2492–2500, Honolulu, HI, 2017.
- [12] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In *CVPR*, pages 3364–3372, Las Vegas, NV, June 2016.
- [13] Eric Brachmann and Carsten Rother. Learning Less is More - 6D Camera Localization via 3D Surface Regression. In *CVPR*, pages 4654–4662, Salt Lake City, UT, June 2018.
- [14] Samarth Brahmabhatt, Jiwwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *CVPR*, pages 2616–2625, Salt Lake City, UT, June 2018.
- [15] Agata Brajdic and Robert Harle. Walk Detection and Step Counting on Unconstrained Smartphones. In *IJCUC*, pages 225–234, September 2013.
- [16] Tobias Briege, Nisha Lakshmana Raichur, Dorsaf Jdidi, Felix Ott, Tobias Feigl, Johannes Rossouw van der Merwe, Alexander Rügamer, and Wolfgang Felber. Multimodal Learning for Reliable Interference Classification in GNSS Signals. In *ION GNSS+*, pages 3210–3234, Denver, CO, September 2022.
- [17] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High Accuracy Optical Flow Estimation Based on a Theory for Warping. In *ECCV*, volume 3024, pages 25–36, 2004.
- [18] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W. Achelik, and Roland Siegwart. The EuRoC Micro Aerial Vehicle Datasets. In *IJRR*, volume 35(10), pages 1157–1163, January 2016.
- [19] Leonid Butyrev, Thorsten Edelhäuser, and Christopher Mutschler. Deep Reinforcement Learning for Motion Planning of Mobile Robots. In *arXiv:1912.09260*, December 2019.
- [20] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. In *T-RO*, volume 37(6), pages 1874–1890, May 2021.
- [21] Nicholas Carlevaris-Bianco, Arash K. Ushani, and Ryan M. Eustice. University of Michigan North Campus Long-Term Vision and Lidar Dataset. In *IJRR*, volume 35(9), December 2015.
- [22] Robert Oliver Castle, Darren J. Gawley, Georg Klein, and David William Murray. Towards Simultaneous Recognition, Localization and Mapping for Hand-Held and Wearable Cameras. In *ICRA*, pages 4102–4107, Rome, Italy, April 2007.
- [23] Moises J. Castro-Toscano, Julio C. Rodríguez-Quiñonez, Daniel Hernández-Balbuena, Lars Lindner, Oleg Sergienko, Moises Rivas-Lopez, and Wendy Flores-Fuentes. A Methodological use of Inertial Navigation Systems for Strapdown Navigation Task. In *ISIE*, pages 1589–1595, Edinburgh, UK, June 2017.
- [24] Changhao Chen, Xiaoxuan Lu, Andrew Markham, and Niki Trigoni. IONet: Learning to Cure the Curse of Drift in Inertial Odometry. In *AAAI Technical Tracks: Robotics*, volume 32(1), January 2018.
- [25] Changhao Chen, Stefano Rosa, Yishu Miao, Chris Xiaoxuan Lu, Wei Wu, Andrew Markham, and Niki Trigoni. Selective Sensor Fusion for Neural Visual-Inertial Odometry. In *CVPR*, pages 10542–10551, Long Beach, CA, June 2019.
- [26] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks. In *ICML*, volume 80, pages 794–803, 2018.
- [27] Ming Chi, Chunhua Shen, and Ian Reid. A Hybrid Probabilistic Model for Camera Relocalization. In *BMVC*, York, UK, 2018.
- [28] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. VidLoc: A Deep Spatio-Temporal Model for 6-DoF Video-Clip Relocalization. In *CVPR*, pages 2652–2660, Honolulu, HI, July 2017.
- [29] Ronald Clark, Sen Wang, Hongkai Wen, Andrew Markham, and Niki Trigoni. VINet: Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem. In *AAAI*, pages 3995–4001, February 2017.
- [30] Gabriele Costante and Thomas Alessandro Ciarfuglia. LS-VO: Learning Dense Optical Subspace for Robust Visual Odometry Estimation. In *RA-L*, volume 3(3), pages 1735–1742, February 2018.
- [31] Gabriele Costante, Michele Mancini, Paolo Valigi, and Thomas A. Ciarfuglia. Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation. In *RA-L*, volume 1(1), pages 18–25, December 2015.
- [32] Anweshan Das and Gijs Dubbelman. An Experimental Study on Relative and Absolute Pose Graph Fusion for Vehicle Localization. In *Intelligent Vehicles Symposium (IV)*, pages 630–635, Changshu, China, June 2018.
- [33] DecaYale. GitHub VLocNet Implementation. In <https://github.com/DecaYale/VLocNet>, 2019.
- [34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, pages 248–255, Miami, FL, June 2009.
- [35] Mingyu Ding, Zhe Wang, Jiankai Sun, Jianping Shi, and Ping Luo. CamNet: Coarse-to-Fine Retrieval for Camera Re-Localization. In *ICCV*, pages 2871–2880, Seoul, South Korea, October 2019.
- [36] João Paulo Silva do Monte Lima, Hideaki Uchiyama, and Rin ichiro Taniguchi. End-to-End Learning Framework for IMU-based 6-DOF Odometry. In *MDPI Sensors*, volume 19(17), page 3777, August 2019.
- [37] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philipp Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *ICCV*, pages 2758–2766, Santiago de Chile, Chile, December 2015.
- [38] Naser El-Sheimy, Haiying Hou, and Xiaoji Niu. Analysis and Modeling of Inertial Sensors Using Allan Variance. In *Trans. on Instrumentation and Measurement*, volume 57(1), pages 140–149, January 2008.
- [39] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct Sparse Odometry. In *TPAMI*, volume 40(3), pages 611–625, April 2017.
- [40] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-Scale Cirect Monocular SLAM. In *ECCV*, volume 8690, pages 834–849, 2014.
- [41] Matthias Faessler, Flavio Fontana, Christian Forster, Elias Mueggler, Matia Pizzoli, and Davide Scaramuzza. Autonomous, Vision-based Flight and Live Dense 3D Mapping with a Quadrotor Micro Aerial Vehicle. In *Journal Field Robotics*, volume 33(4), pages 431–450, June 2016.
- [42] Zheyu Feng, Jianwen Li, Luhong Zhang, and Chen Chen. Online Spatial and Temporal Calibration for Monocular Direct Visual-Inertial Odometry. In *MDPI Sensors*, volume 19(10), May 2019.
- [43] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems. In *T-RO*, volume 33(2), pages 249–265, April 2017.
- [44] Yarin Gal and Zoubin Ghahramani. Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. In *arXiv:1506.02158*, June 2015.
- [45] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete Solution Classification for the Perspective-Three-Point Problem. In *TPAMI*, pages 930–943, August 2003.
- [46] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision Meets Robotics: The KITTI Dataset. In *IJRR*, volume 32(11), pages 1231–1237, August 2013.
- [47] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *CVPR*, pages 3354–3361, Providence, RI, June 2012.

[48] Puneet Goel, Stergios I. Roumeliotis, and Gaurav S. Sukhatme. Robust Localization using Relative and Absolute Position Estimates. In *Intl. Conf. on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients*, volume 2, pages 1134–1140, Kyongju, South Korea, October 1999.

[49] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A Novel Connectionist System for Unconstrained Handwriting Recognition. In *TPAMI*, volume 31(5), pages 855–868, May 2009.

[50] Matthew Koichi Grimes, Dragomir Anguelov, and Yann LeCun. Hybrid Hessians for Flexible Optimization of Pose Graphs. In *Intl. Conf. on Intelligent Robots and Systems*, Taipei, Taiwan, October 2010.

[51] Dai-In Han, Timothy Jung, and Alex Gibson. Dublin AR: Implementing Augmented Reality in Tourism. In *ICCT*, pages 511–523, 2013.

[52] Liming Han, Yimin Lin, Guoguang Du, and Shiguo Lian. DeepVIO: Self-supervised Deep Learning of Monocular Visual Inertial Odometry using 3D Geometric Constraints. In *IROS*, Macau, China, November 2019.

[53] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. In *Neural Computation*, volume 9(8), pages 1735–1780, November 1997.

[54] Euntae Hong and Jongwoo Lim. Visual-Inertial Odometry with Robust Initialization and Online Scale Estimation. In *MDPI Sensors*, volume 18(12), page 4287, December 2018.

[55] Berthold K. P. Horn. Closed-form Solution of Absolute Orientation Using Unit Quaternions. In *Journal of the Optical Society of America*, volume 4(4), pages 629–642, April 1987.

[56] Di Hu, Chengze Wang, Feiping Nie, and Xuelong Li. Dense Multimodal Fusion for Hierarchically Joint Representation. In *ICASSP*, pages 3941–3945, Brighton, UK, May 2019.

[57] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks, 7132–7141. In *(CVPR)*, 2018.

[58] Jwu-Sheng Hu and Ming-Yuan Chen. A Sliding-Window Visual-IMU Odometer Based on Tri-Focal Tensor Geometry. In *ICRA*, pages 3963–3968, Hong Kong, China, May 2014.

[59] Zhaoyang Huang, Yan Xu, Jianping Shi, Xiaowei Zhou, Hujun Bao, and Guofeng Zhang. Prior Guided Dropout for Robust Visual Localization in Dynamic Environments. In *ICCV*, pages 2791–2800, Seoul, Korea, October 2019.

[60] Hyon Lim and Sudipta N. Sinha and Michael F. Cohen and Matthew Uyttendaele. Real-time monocular image-based 6-dof localization in large-scale environments. In *CVPR*, pages 476–492, Providence, RI, June 2015.

[61] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *CVPR*, pages 1647–1655, Honolulu, HI, July 2017.

[62] Ganesh Iyer, J. Krishna Murthy, Gunshi Gupta, K. Madhava Krishna, and Liam Paull. Geometric Consistency for Self-Supervised End-to-End Visual Odometry. In *CVPRW*, Salt Lake City, UT, June 2018.

[63] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement Learning with Unsupervised Auxiliary Tasks. In *arXiv:1611.05397*, November 2016.

[64] Gijeong Jang, Sungho Lee, and Inso Kweon. Color Landmark Based Self-Localization for Indoor Mobile Robots. In *ICRA*, pages 1037–1042, Washington, DC, May 2002.

[65] Hamid Reza Vaezi Jozé, Amirreza Shaban, Michael L. Iuzzolino, and Kazuhito Koishida. MMTM: Multimodal Transfer Module for CNN Fusion. In *CVPR*, pages 13289–13299, Seattle, WA, June 2020.

[66] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John Leonard, and Frank Dellaert. iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree. In *IJRR*, pages 216–235, Shanghai, China, May 2012.

[67] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-Scale Video Classification with Convolutional Neural Networks. In *CVPR*, pages 1725–1732, Columbus, OH, June 2014.

[68] Anton Kasyanov, Francis Engelmann, Jörg Stückler, and Bastian Leibe. Keyframe-based Visual-Inertial Online SLAM with Relocalization. In *IROS*, pages 6662–6669, Vancouver, Canada, September 2017.

[69] Alex Kendall and Roberto Cipolla. Modelling Uncertainty in Deep Learning for Camera Relocalization. In *ICRA*, pages 4762–4769, Stockholm, Sweden, May 2016.

[70] Alex Kendall and Roberto Cipolla. Geometric Loss Functions for Camera Pose Regression with Deep Learning. In *CVPR*, pages 6555–6564, Honolulu, HI, July 2017.

[71] Alex Kendall and Yarin Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *NIPS*, pages 5580–5590, December 2017.

[72] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *ICCV*, pages 2938–2946, Santiago de Chile, Chile, December 2015.

[73] Christian Kerl, Jürgen Sturm, and Daniel Cremers. Dense Visual SLAM for RGB-D Cameras. In *IROS*, pages 2100–2106, Tokyo, Japan, November 2013.

[74] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *ISMAR*, pages 225–234, Nara, Japan, November 2007.

[75] Robin Kreuzig, Matthias Ochs, and Rudolf Mester. DistanceNet: Estimating Traveled Distance From Monocular Images Using a Recurrent Convolutional Neural Network. In *CVPRW*, Long Beach, CA, June 2019.

[76] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera Relocalization by Computing Pairwise Relative Poses Using Convolutional Neural Network. In *ICCVW*, pages 920–929, Venice, Italy, 2017.

[77] Sooyong Lee and Jae-Bok Song. Mobile Robot Localization Using Infrared Light Reflecting Landmarks. In *Intl. Conf. Control, Automation and Systems*, pages 674–677, Seoul, South Korea, October 2007.

[78] Stefan Leutenegger, Paul Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization. In *IJRR*, volume 34(3), pages 314–334, 2015.

[79] Kumpeng Li, Ziyian Wu, Kuan-Chuan Peng, Jan Ernst, and Yun Fu. Tell Me Where to Look: Guided Attention Inference Network. In *CVPR*, pages 9215–9223, Salt Lake City, UT, June 2018.

[80] Peiliang Li, Tong Qin, Botao Hu, Fengyuan Zhu, and Shaojie Shen. Monocular Visual-Inertial State Estimation for Mobile Augmented Reality. In *ISMAR*, Nantes, France, October 2017.

[81] Wenbin Li, Sajad Saeedi, John McCormac, Ronald Clark, Dimos Tsoumanikas, Qing Ye, Yuzhong Huang, Rui Tang, and Stefan Leutenegger. InteriorNet: Mega-Scale Multi-Sensor Photo-Realistic Indoor Scenes Dataset. In *BMVC*, volume 18(17), 2018.

[82] Xin Li, Xingtai Ou, Zhi Li, Hengl Wei, Wei Zhou, and Zhemin Duan. On-Line Temperature Estimation for Noisy Thermal Sensors Using a Smoothing Filter-Based Kalman Predictor. In *Sensors*, volume 18(2), page 433, February 2018.

[83] Yunpeng Li, Noah Snavely, and Daniel P. Huttenlocher. Location Recognition Using Prioritized Feature Matching. In *ECCV*, pages 791–804, 2010.

[84] Lukas Liebel and Marco Körner. Auxiliary Tasks in Multi-Task Learning. In *arXiv:1805.06334*, May 2018.

[85] Yimin Lin, Xiangxiong Liu, Jianfeng Huang, Chaopeng Wang, Guoguang Du, Jingjiang Bai, Shiguo Lian, and Bill Huang. Deep Global-Relative Networks for End-to-End 6-DoF Visual Localization and Odometry. In *PRICAI*, pages 454–467, Cuvu, Fiji, 2019.

[86] Haomin Liu, Mingyu Chen, Guofeng Zhang, Hujun Bao, and Yingze Bao. ICE-BA: Incremental, Consistent and Efficient Bundle Adjustment for Visual-Inertial SLAM. In *CVPR*, pages 1974–1982, Salt Lake City, UT, June 2018.

[87] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-End Multi-Task Learning with Attention. In *CVPR*, pages 1871–1880, Long Beach, CA, June 2019.

[88] Christoffer Löffler, Sascha Riechel, Janina Fischer, and Christopher Mutschler. Evaluation Criteria for Inside-Out Indoor Positioning Systems Based on Machine Learning. In *IPIN*, pages 1–8, Nantes, France, September 2018.

[89] Simon Lynen, Torsten Sattler, Michael Bosse, Joel A. Hesch, Marc Pollefeys, and Roland Siegwart. Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization. In *Robotics: Science and Systems*, volume 39(9), July 2020.

[90] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. In *IJRR*, volume 36(1), pages 3–15, November 2016.

[91] Niklas Magnusson and Tobias Odenman. Improving Absolute Position Estimates of an Automotive Vehicle using GPS in Sensor Fusion. In *Chalmers University of Technology, Thesis*, Göteborg, Sweden, 2012.

[92] Syaiful Mansur, Muhammad Habib, Gilang Nugraha Putu Pratama, Adha Imani Cahyadi, and Igi Ardijanto. Real Time Monocular Visual Odometry using Optical Flow: Study on Navigation of Quadrotora's UAV. In *Intl. Conf. on Science and Technology - Computer (ICST)*, pages 122–126, Yogyakarta, Indonesia, July 2017.

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

318

AUGUST 2023

19

[93] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose Estimation for Augmented Reality: A Hands-On Survey. In *TVCG*, volume 22(12), pages 2633–2651, December 2016.

[94] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Image-Based Localization Using Hourglass Networks. In *ICCVW*, pages 870–877, Venice, Italy, October 2017.

[95] Sherif A. S. Mohamed, Mohammad-Hashem Haghbayan, Tomi Westerlund, Jukka Heikkonen, Hannu Tenhunen, and Juha Plosila. A Survey on Odometry for Autonomous Navigation Systems. In *IEEE Access*, volume 7, pages 97466–97486, July 2019.

[96] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanculescu, and Arnaud de La Fortelle. CoordiNet: Uncertainty-aware Pose Regressor for Reliable Vehicle Localization. In *WACV*, January 2022.

[97] Anastasios I. Mourikis and Stergios I. Roumeliotis. A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation. In *ICRA*, pages 3565–3572, Roma, Italy, April 2007.

[98] Xufu Mu, Jing Chen, Zixiang Zhou, Zhen Leng, and Lei Fan. Accurate Initial State Estimation in a Monocular Visual-Inertial SLAM System. In *MDPI Sensors*, volume 18(2), page 506, February 2018.

[99] Peter Müller and Andreas Savakis. Flowdometry: An Optical Flow and Deep Learning Based Approach to Visual Odometry. In *WACV*, volume 1, pages 624–631, Santa Rosa, CA, 2017.

[100] Peter M. Müller, Andreas Savakis, Raymond Ptucha, and Roy Melton. Optical Flow and Deep Learning Based Approach to Visual Odometry. In *Rochester Institute of Technology, Department of Computer Engineering*, Rochester, NY, November 2016.

[101] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. In *T-RO*, volume 31(5), pages 1147–1163, 2015.

[102] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. In *T-RO*, volume 33(5), pages 1255–1262, June 2017.

[103] Raúl Mur-Artal and Juan D. Tardós. Visual-Inertial Monocular SLAM With Map Reuse. In *RA-L*, volume 2(2), pages 796–803, January 2017.

[104] Tayyab Naseer and Wolfram Burgard. Deep Regression for Monocular Camera-based DoF Global Localization in Outdoor Environments. In *IROS*, pages 1525–1530, Vancouver, BC, September 2017.

[105] Pradeep Natarajan, Shuang Wu, Shiv Vitaladevuni, Xiaodan Zhuang, Stavros Tsakalidis, Unsang Park, Rohit Prasad, and Premkumar Natarajan. Multimodal Feature Fusion for Robust Event Detection in Web Videos. In *CVPR*, pages 1298–1305, Providence, RI, June 2012.

[106] Aviv Navon, Idan Achituv, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary Learning by Implicit Differentiation. In *ICLR*, 2021.

[107] David Nister, Oleg Naroditsky, and James Bergen. Visual Odometry. In *CVPR*, Washington, DC, June 2004.

[108] Gabriel Nützi, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Fusion of IMU and Vision for Absolute Scale Estimation in Monocular SLAM. In *Journal of Intelligent and Robotic Systems*, volume 61, pages 287–299, 2011.

[109] Felix Ott, Tobias Feigl, Christoffer Löffler, and Christopher Mutschler. ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization. In *CVPRW*, pages 187–198, Seattle, WA, June 2020.

[110] Andrew Owens and Alexei A. Efros. Audio-Visual Scene Analysis with Self-Supervised Multisensory Features. In *ECCV*, pages 631–648, 2018.

[111] Mitesh Patel, Brendan Emery, and Yan-Ying Chen. ContextualNet: Exploiting Contextual Information using LSTMs to Improve Image-based Localization. In *ICRA*, Brisbane, Australia, May 2018.

[112] Bernd Pfommer, Nitin Sanket, Kostas Daniilidis, and Jonas Cleveland. PennCOSYVIO: A Challenging Visual Inertial Odometry Benchmark. In *ICRA*, pages 3847–3854, Singapore, Singapore, May 2017.

[113] Nathan Piasco, Désiré Sidibé, Valérie Gouet-Brunet, and Cédric Demonceaux. Improving Image Description with Auxiliary Modality for Visual Localization in Challenging Conditions. In *IJCV*, volume 129(1), pages 185–202, August 2020.

[114] Tong Qin, Peiliang Li, and Shaojie Shen. Relocalization, Global Optimization and Map Merging for Monocular Visual-Inertial SLAM. In *ICRA*, Brisbane, Australia, May 2018.

[115] Tong Qin, Peiliang Li, and Shaojie Shen. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. In *T-RO*, volume 34(4), pages 1004–1020, July 2018.

[116] Tong Qin, Jie Pan, Shaozhi Cao, and Shaojie Shen. A General Optimization-based Framework for Local Odometry Estimation with Multiple Sensors. In *arXiv:1901.03638*, January 2019.

[117] Noha Radwan, Abhinav Valada, and Wolfram Burgard. VLocNet++: Deep Multitask Learning for Semantic Visual Localization and Odometry. In *RA-L*, volume 3(4), pages 4407–4414, 2018.

[118] Nisha Lakshmana Raichur. Image-/IMU-based Deep Multimodal Information Fusion for Pedestrian Self-Localization. In *Master's Thesis, University of Ulm*, April 2021.

[119] Rebecca L. Russell and Christopher Reale. Multivariate Uncertainty in Deep Learning. In *Trans. on Neural Networks and Learning Systems*, pages 1–7, June 2021.

[120] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization. In *TPAMI*, volume 39(9), pages 1744–1756, September 2017.

[121] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In *CVPR*, Salt Lake City, UT, June 2018.

[122] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixé. Understanding the Limitations of CNN-based Absolute Camera Pose Regression. In *CVPR*, pages 3302–3312, Long Beach, CA, June 2019.

[123] David Schubert, Thore Goll, Nikolaus Demmel, Vladyslav Usenko, Jörg Stücker, and Daniel Cremers. The TUM VI Benchmark for Evaluating Visual-Inertial Odometry. In *IROS*, Madrid, Spain, October 2018.

[124] Soroush Seifi and Tinne Tuytelaars. How to improve CNN-based 6-DoF Camera Pose Estimation. In *ICCVW*, Seoul, Korea, October 2019.

[125] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *CVPR*, pages 2930–2937, Portland, OR, June 2013.

[126] Yuanchao Shu, Kang G. Shin, Tian He, and Jiming Chen. Last-Mile Navigation Using Smartphones. In *Intl. Conf. on Mobile Computing and Networking (MCN)*, pages 512–524, September 2015.

[127] Gabe Sibley, Larry Matthies, and Gaurav Sukhatme. Sliding Window Filter with Application to Planetary Landing. In *Journal of Field Robotics (JFR)*, pages 587–608, August 2010.

[128] Randall C. Smith and Peter Cheeseman. On the Representation and Estimation of Spatial Uncertainty. In *IJRR*, volume 5(4), pages 56–68, 1986.

[129] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which Tasks Should be Learned Together in Multi-Task Learning? In *ICML*, volume 119, pages 9120–9132, 2020.

[130] Christopher Stapleton, Charles Hughes, Michael Moshell, Paulius Mičikevičius, and Marty Altman. Applying Mixed Reality to Entertainment. In *IEEE*, volume 35(12), pages 122–124, December 2002.

[131] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *IROS*, October 2012.

[132] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. In *CVPR*, pages 1–9, Boston, MA, 2015.

[133] Takahiro Terashima and Osamu Hasegawa. A Visual-SLAM for First Person Vision and Mobile Robots. In *MVA*, Nagoya, Japan, May 2017.

[134] Abhinav Valada, Noha Radwan, and Wolfram Burgard. Deep Auxiliary Learning for Visual Localization and Odometry. In *ICRA*, pages 6939–6946, Brisbane, Australia, may 2018.

[135] Julien Valentin, Angela Dai, Matthias Niessner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to Navigate the Energy Landscape. In *3DV*, Stanford, CA, October 2016.

[136] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *NIPS*, 2017.

[137] Francesco Visin, Kyle Kastner, Kyunghyun Cho, Matteo Matteucci, Aaron Courville, and Yoshua Bengio. ResNet: A Recurrent Neural Network Based Alternative to Convolutional Networks. In *arXiv:1505.00393*, May 2015.

[138] Lukas von Stumberg, Vladyslav Usenko, and Daniel Cremers. Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization. In *ICRA*, Brisbane, Australia, May 2018.

[139] Lukas von Stumberg, Patrick Wenzel, Nan Yang, and Daniel Cremers. LM-Reloc: Levenberg-Marquardt Based Direct Visual Relocalization. In *3DV*, Fukuoka, Japan, November 2020.

[140] Florian Walch, Caner Hazirbas, Laura Leal-Taixé, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-Based Localization Using LSTMs for Structured Feature Correlation. In *ICCV*, pages 627–637, Venice, Italy, October 2017.

- [141] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. AtLoc: Attention Guided Camera Localization. In *AAAI*, volume 34(06), pages 10393–10401, April 2020.
- [142] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. DeepVO: Towards end-to-end Visual Odometry with deep Recurrent Convolutional Neural Networks. In *ICRA*, pages 2043–2050, Singapore, Singapore, June 2017.
- [143] Brian Williams, Georg Klein, and Ian Reid. Real-time SLAM Relocalisation. In *ICCV*, pages 1–8, Rio de Janeiro, Brazil, October 2007.
- [144] Jian Wu, Liwei Ma, and Xiaolin Hu. Delving Deeper Into Convolutional Neural Networks for Camera Relocalization. In *ICRA*, pages 5644–5651, Singapore, Singapore, May 2017.
- [145] Binbin Xu, Andrew J. Davison, and Stefan Leutenegger. Deep Probabilistic Feature-Metric Tracking. In *RA-L*, volume 6(1), pages 223–230, January 2021.
- [146] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, volume 37, pages 2048–2057, 2015.
- [147] Xuejiao Yan, Zongying Shi, and Yisheng Zhong. Vision-based Global Localization of Unmanned Aerial Vehicles with Street View Images. In *CCC*, pages 4672–4678, Wuhan, China, July 2018.
- [148] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. D3VO: Deep Depth, Deep Pose and Deep Uncertainty for Monocular Visual Odometry. In *CVPR*, pages 1281–1292, Seattle, WA, June 2020.
- [149] Nan Yang, Rui Wang, Xiang Gao, and Daniel Cremers. Challenges in Monocular Visual Odometry: Photometric Calibration, Motion Bias, and Rolling Shutter Effect. In *RA-L*, volume 3(4), pages 2878–2885, June 2018.
- [150] Aurélien Yol, Bertrand Delabarre, Amaury Dame, Jean Émile Dartois, and Eric Marchand. Vision-based Absolute Localization for Unmanned Aerial Vehicles. In *ICIRS*, pages 3429–3434, Chicago, IL, September 2014.
- [151] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera Pose Voting for Large-Scale Image-Based Localization. In *ICCV*, pages 2704–2712, Santiago de Chile, Chile, December 2015.
- [152] Huangying Zhan, Chamara Saroj Weerasekera, Jia-Wang Bian, and Ian Reid. Visual Odometry Revisited: What Should be Learnt? In *ICRA*, pages 4203–4210, Paris, France, September 2020.
- [153] Qi Zhao, Fangmin Li, and Xinhua Liu. Real-time Visual Odometry based on Optical Flow and Depth Learning. In *ICMTMA*, pages 239–242, Changsha, China, February 2018.
- [154] Lei Zhou, Zixin Luo, Tianwei Shen, Jiahui Zhang, Mingmin Zhen, Yao Yao, Tian Fang, and Long Quan. KFNet: Learning Temporal Camera Relocalization using Kalman Filtering. In *CVPR*, pages 4919–4928, Seattle, WA, June 2020.
- [155] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the Continuity of Rotation Representations in Neural Networks. In *CVPR*, pages 5738–5746, Long Beach, CA, June 2019.
- [156] David Zuñiga-Noël, Alberto Jaenal, Ruben Gomez-Ojeda, and Gonzalez-Jimenez. The UMA-VI Dataset: Visual-Inertial Odometry in Low-Textured and Dynamic Illumination Environments. In *IJRR*, volume 39(9), July 2020.

BIOGRAPHY



Felix Ott received his M.Sc. degree in Computational Engineering at the Friedrich-Alexander-Universität (FAU) Erlangen-Nürnberg, Germany, in 2019. He joined the Hybrid Positioning & Information Fusion group in the Locating and Communication Systems department at Fraunhofer IIS. In 2020, he started his Ph.D. at the Ludwig-Maximilians-Universität (LMU) in Munich in the Probabilistic Machine and Deep Learning group. His research covers multimodal information fusion for self-localization.



Nisha Lakshmana Raichur received her M.Sc. degree in Cognitive Systems at the Technical University of Ulm in 2021. She joined the Hybrid Positioning & Information Fusion group in the Localization and Communication Systems department at Fraunhofer IIS. Her research covers the areas of machine learning and deep multi-modal sensor fusion.



David Rügamer is an interim professor for Computational Statistics at the TU Dortmund. Before he was an interim professor at the RWTH Aachen and interim professor for Data Science at the LMU Munich, where he also received his Ph.D. in 2018. His research is concerned with scalability of statistical modeling as well as machine learning for functional and multimodal data.



Tobias Feigl is a research associate at the Fraunhofer Institute for Integrated Circuits (IIS) in Nürnberg, and received his Ph.D. at the FAU Erlangen-Nürnberg. His research covers the areas of augmented and virtual reality, human-computer interaction, and machine learning. He improves human motion behavior in immersive virtual environments with inertial and radio sensors.



Heiko Neumann is a full professor at the university of Ulm. He is heading the vision and perception science group focusing on the investigation of mechanisms and the underlying structure of visual information processing in biological and technical systems as well as their adaptation to changing environments. The topics of his investigation primarily focus on mechanisms of visual information processing, namely biological and machine vision.



Bernd Bischl is a full professor for statistical learning and data science at the LMU Munich and a director of the Munich Center of Machine Learning. His research focuses amongst other things on AutoML, interpretable machine learning and machine learning benchmarking.



Christopher Mutschler leads the precise positioning and analytics department at Fraunhofer IIS. Prior to that, Christopher headed the Machine Learning & Information Fusion group. He gives lectures on machine learning at the FAU Erlangen-Nürnberg, from which he also received both his Diploma and Ph.D. in 2010 and 2014 respectively. Christopher's research combines machine learning with radio-based localization.

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

AUGUST 2023

21

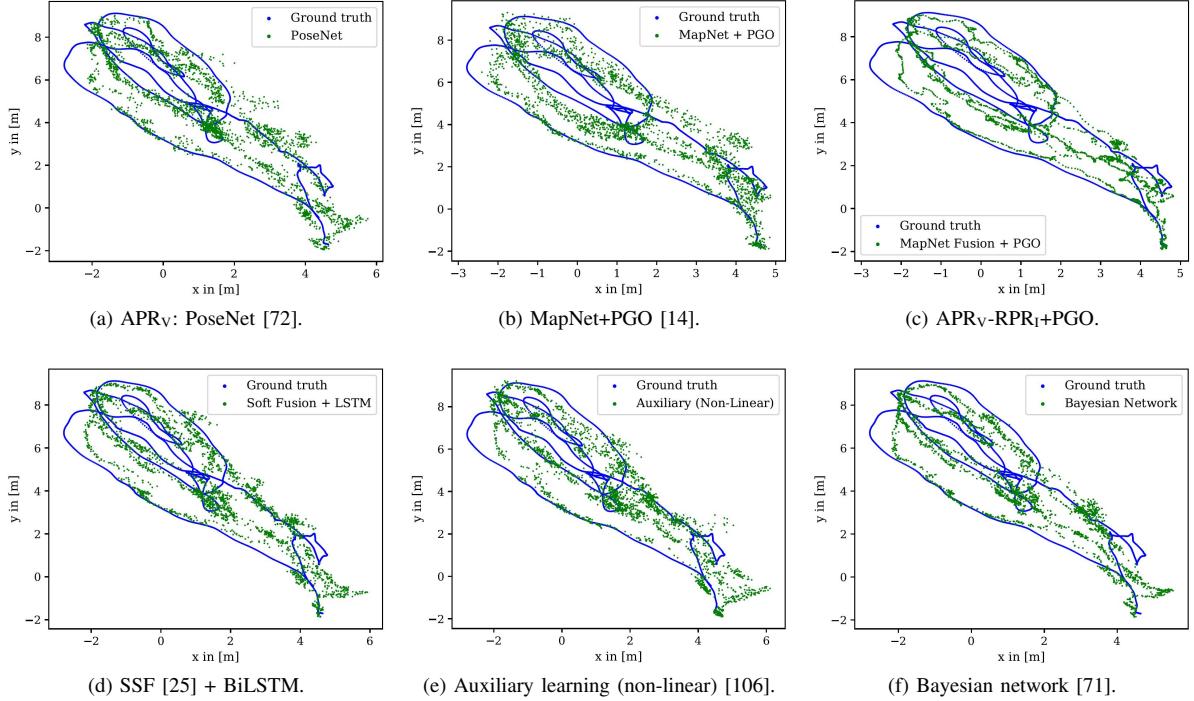


Fig. 22: APRv-RPR_i fusion on EuRoC MAV [18]: MH-02-easy.

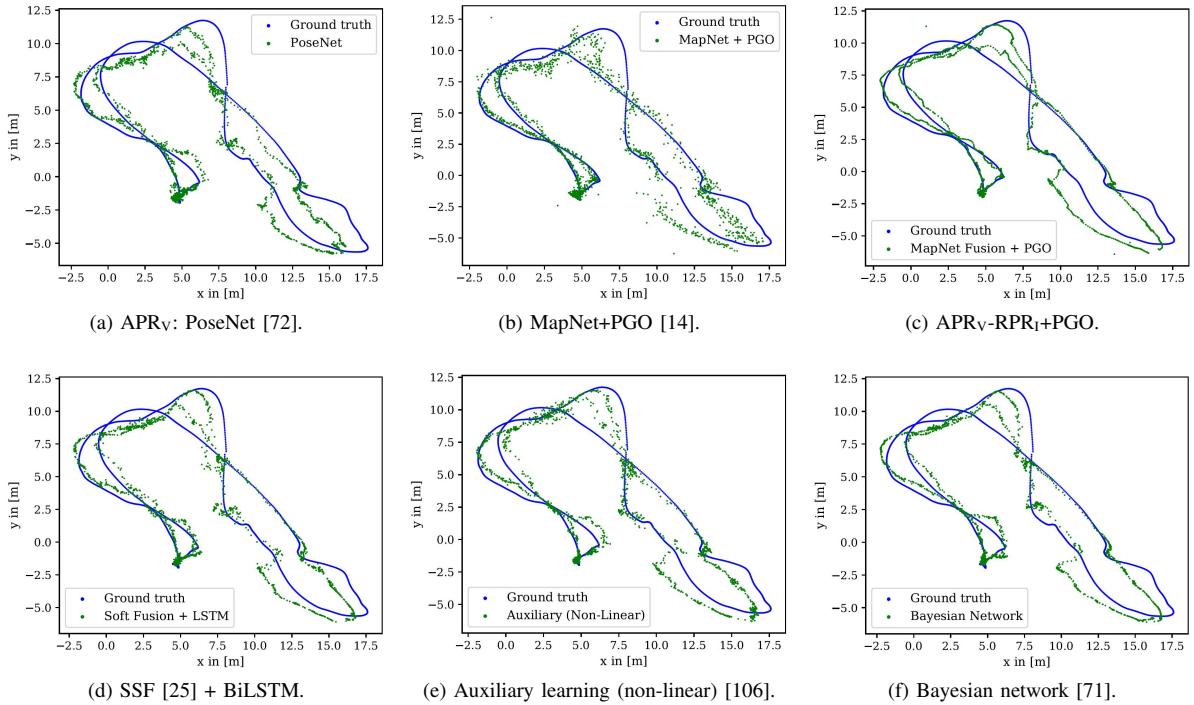
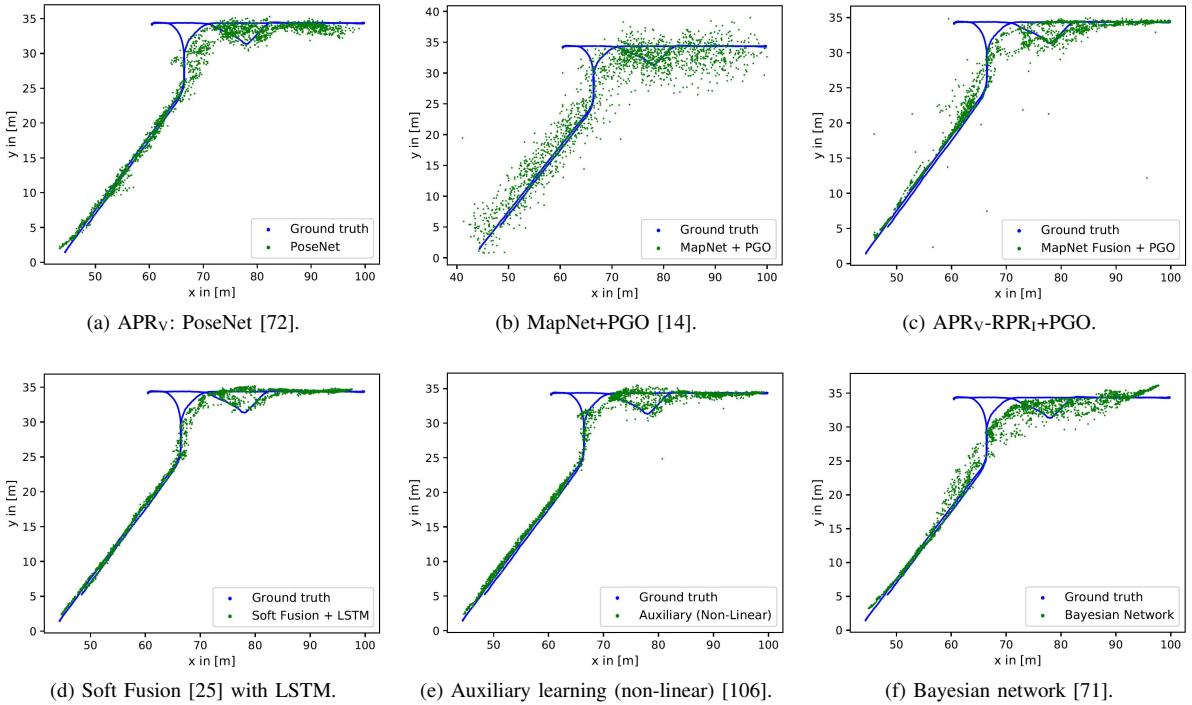
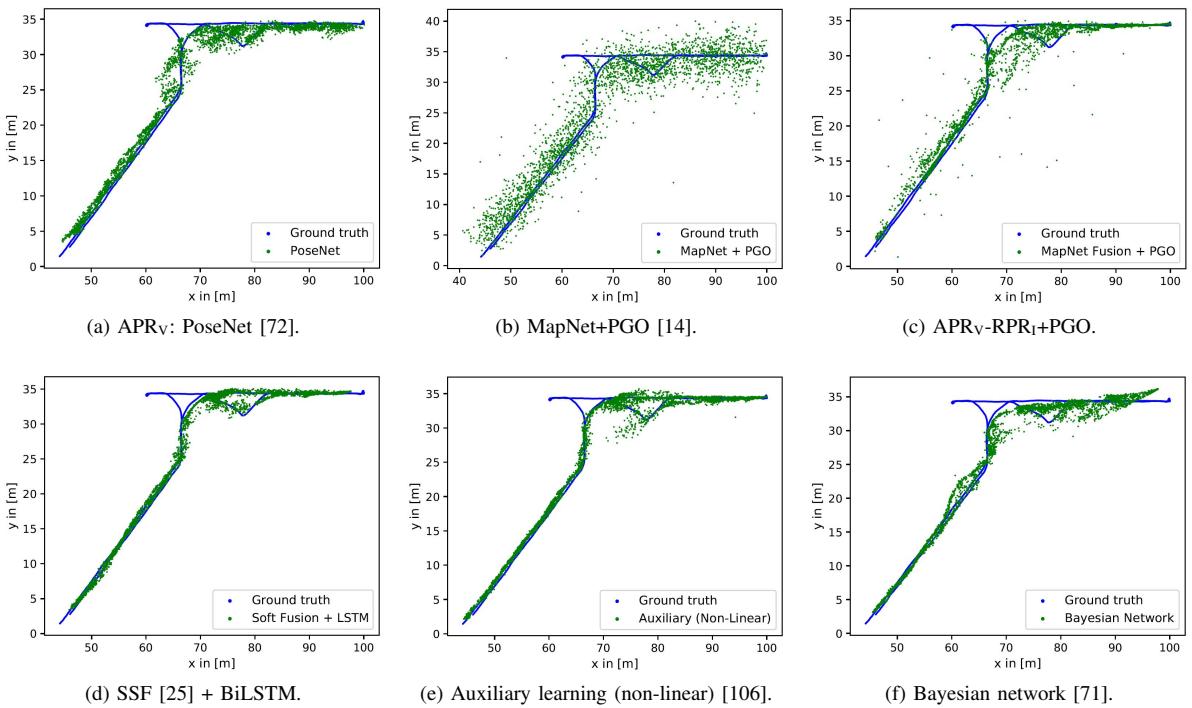


Fig. 23: APRv-RPR_i fusion on EuRoC MAV [18]: MH-04-difficult.

Fig. 24: APR_v-RPR_i fusion on PennCOSYVIO [112]: BF.Fig. 25: APR_v-RPR_i fusion on PennCOSYVIO [112]: BS.

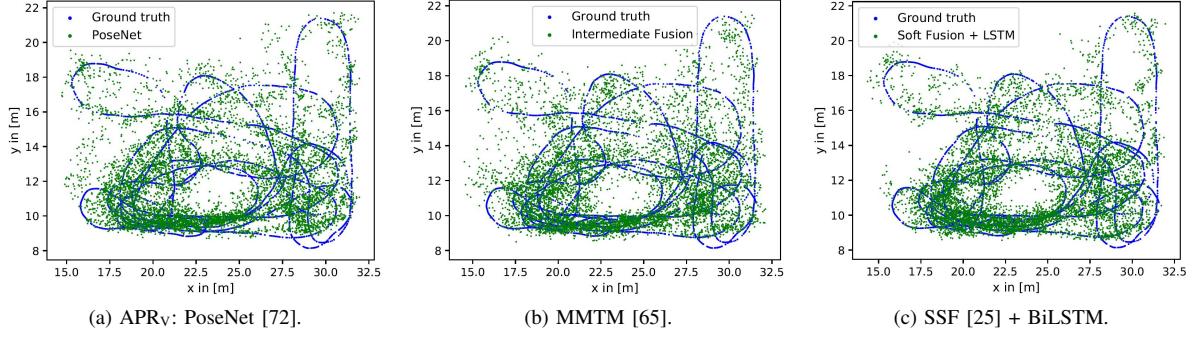


Fig. 26: APRv-RPRi fusion on IndustyVI: Testing trajectory 1.

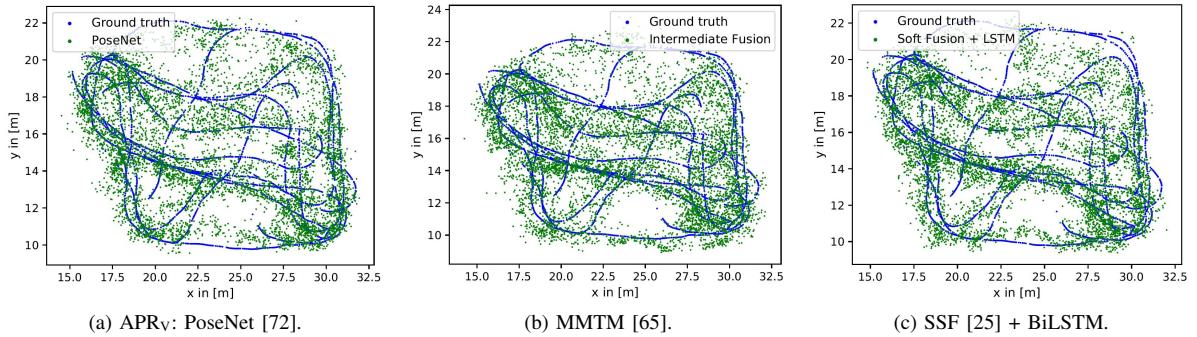


Fig. 27: APRv-RPRi fusion on IndustyVI: Testing trajectory 2.

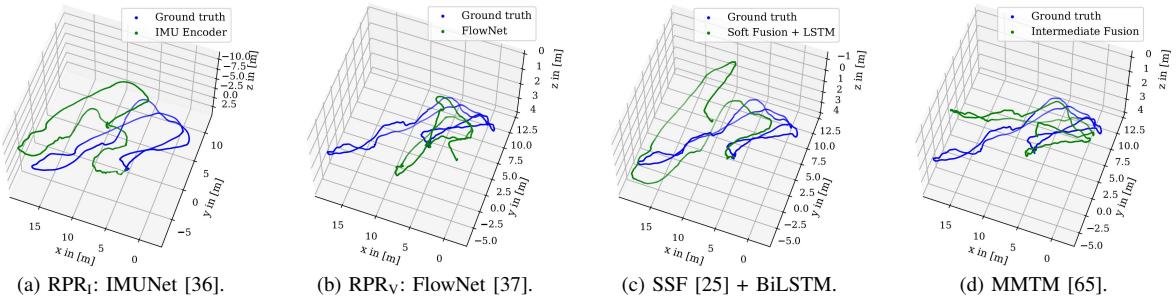
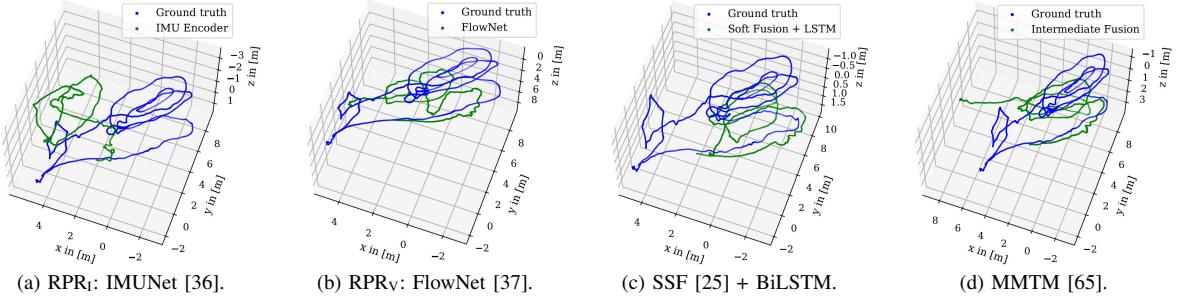
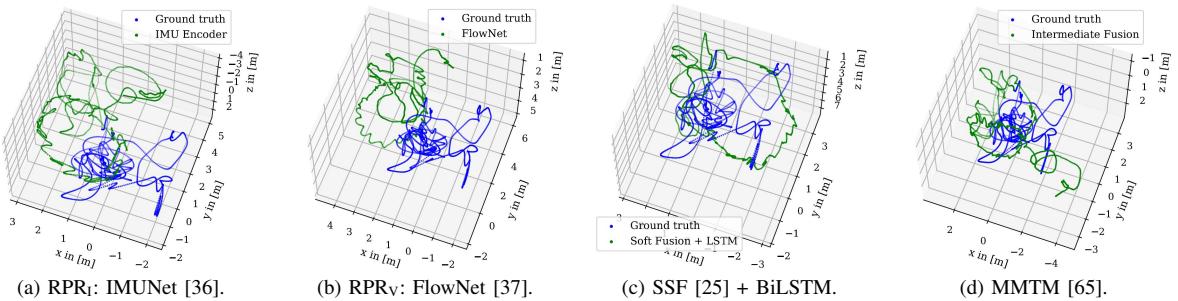
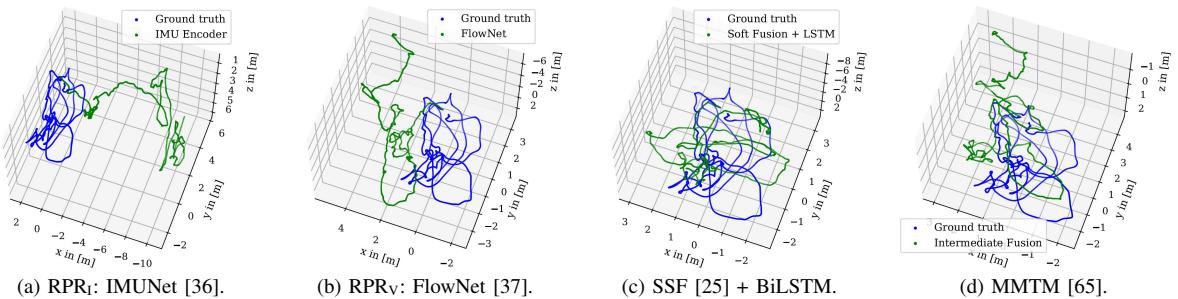
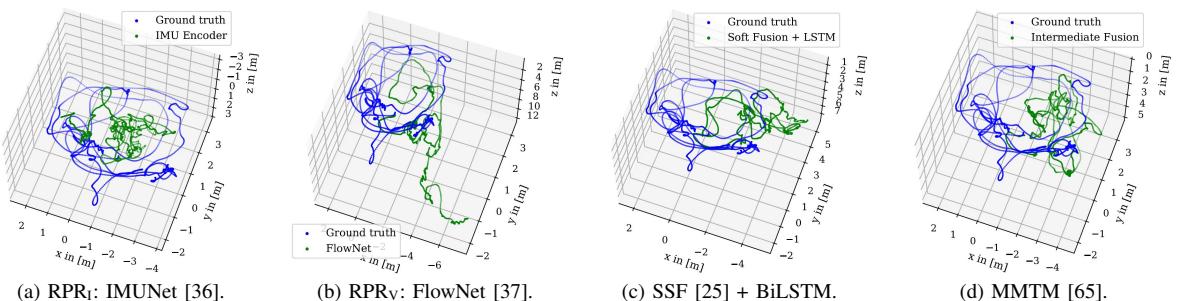


Fig. 28: RPRv-RPRi fusion on EuRoC MAV [18]: MH-04-difficult.

Fig. 29: RPR_V-RPR_I fusion on EuRoC MAV [18]: MH-01-easy.Fig. 30: RPR_V-RPR_I fusion on EuRoC MAV [18]: V1-03-difficult.Fig. 31: RPR_V-RPR_I fusion on EuRoC MAV [18]: V1-01-easy.Fig. 32: RPR_V-RPR_I fusion on EuRoC MAV [18]: V2-02-Medium.

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

324

AUGUST 2023

25

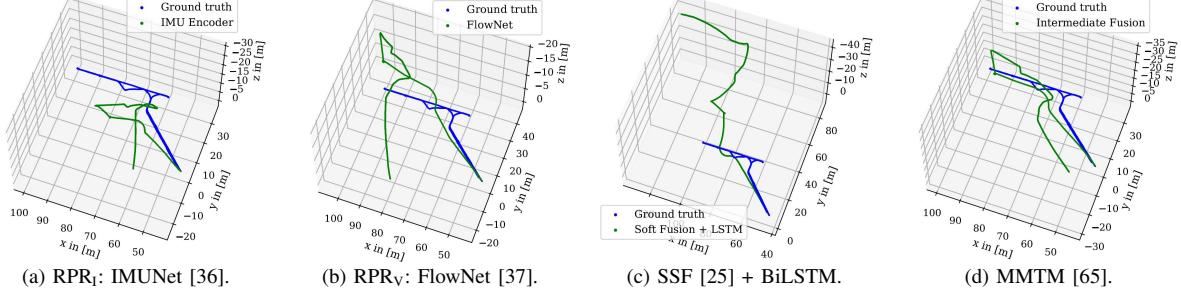


Fig. 33: RPR_V-RPR_I fusion on PennCOSYVIO [112]: BF.

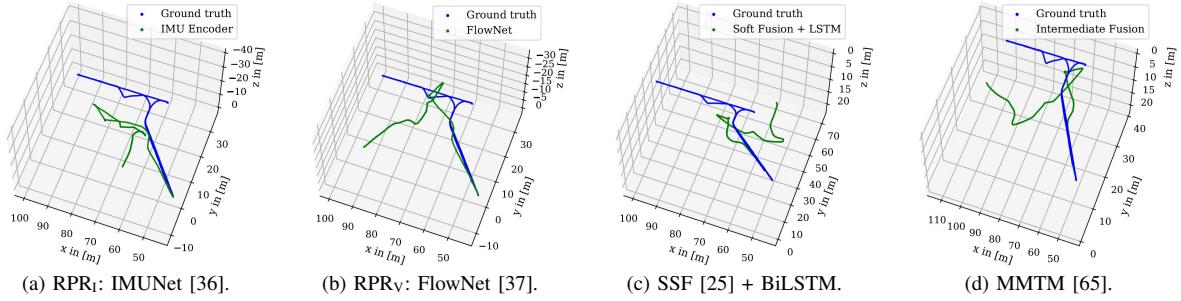


Fig. 34: RPR_V-RPR_I fusion on PennCOSYVIO [112]: BS.

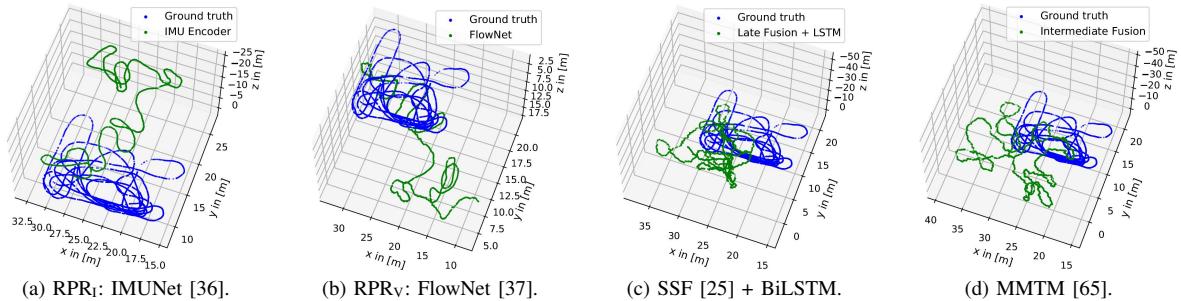


Fig. 35: RPR_V-RPR_I fusion on IndustryVI: Testing Sequence 1.

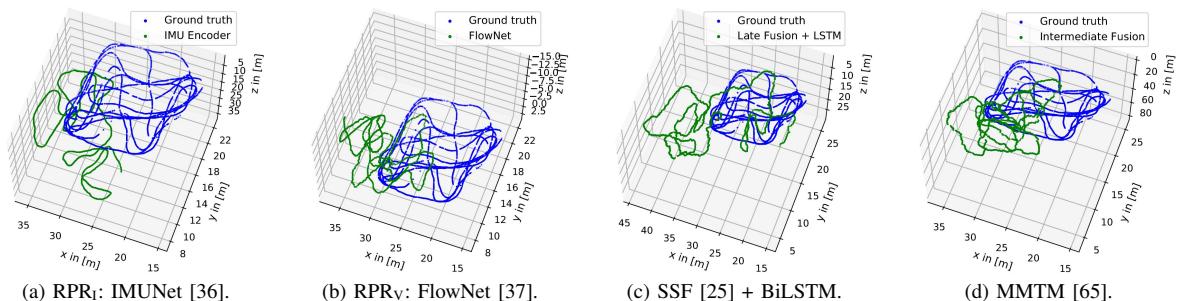


Fig. 36: RPR_V-RPR_I fusion on IndustryVI: Testing Sequence 2.

TABLE VIII: We summarize the number of trainable parameters for the baseline and fusion models. While “F” indicates the layer with MMTM fusion (see Section III-C3), “–” indicates no fusion. While the IMUNet is a small network with 961,031 parameters, PoseNet (7.7 million) and FlowNet (16.7 million) are significantly larger in size and require more computing time. The number of parameters decreases by combining the APR_V or RPR_V models with the RPR_I model for the late fusion, as the last model layers are removed. By adding BiLSTM layers, the model size increases.

Method	# Parameters	# Parameters
Baseline Models		
APR _V (PoseNet)	7,713,447	-
RPR _I (IMUNet)	961,031	961,031
RPR _V (FlowNet)	-	16,715,520
Method	# Parameters	# Parameters
	APR _V -RPR _I	RPR _V -RPR _I
Fusion Models		
MapNet+PGO	7,713,447	-
APR _V -RPR _I +PGO	7,304,238	-
Late Fusion (concat)	6,726,830	15,654,727
Late Fusion (concat) + BiLSTM	7,304,238	16,315,079
Late Fusion (SSF)	6,792,622	15,671,239
Late Fusion (SSF) + BiLSTM	7,320,750	16,331,591
Intermediate Fusion:		
MMTM (F/–/–)	8,489,614	16,906,055
MMTM (–/F/–)	8,489,614	16,906,055
MMTM (–/–/F)	8,944,558	17,955,399
MMTM (F/F/–)	9,674,990	17,497,031
MMTM (–/F/F)	10,129,934	18,546,375
MMTM (F/–/F)	10,129,934	18,546,375
MMTM (F/F/F)	11,315,310	19,137,351
Auxiliary (non-linear)	7,320,769	-
Auxiliary (convolutional)	7,320,948	-
Bayesian network	7,320,750	16,322,453

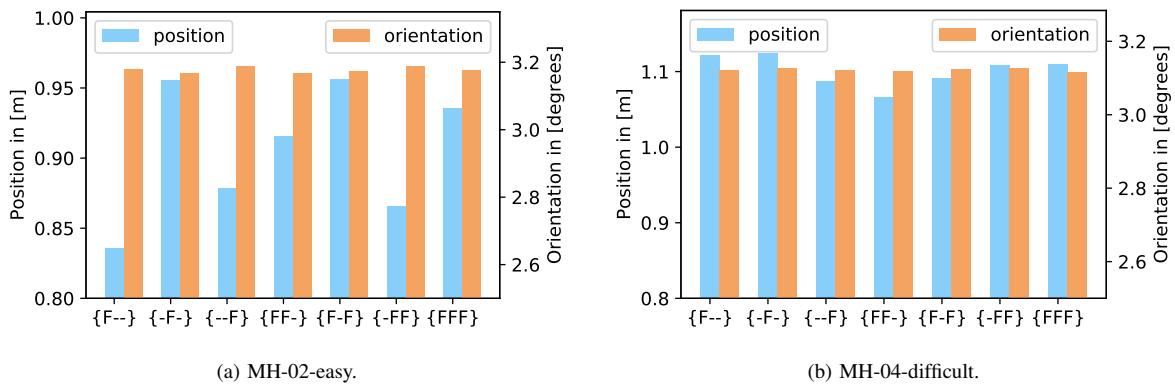


Fig. 37: Evaluation of various fusion combinations of MMTM modules for the EuRoC MAV [18] dataset. While “F” indicates the layer with MMTM fusion (see Section III-C3), “–” indicates no fusion.

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

AUGUST 2023

27

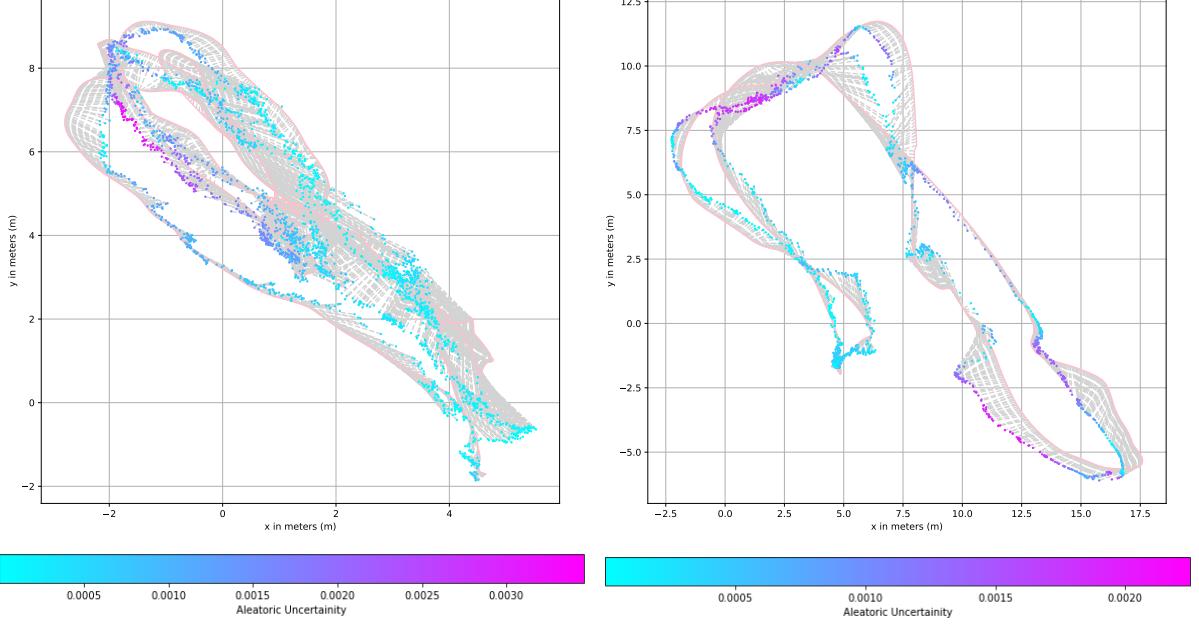


Fig. 38: Plot of the aleatoric uncertainty for the absolute pose prediction. We plot a dashed line from the predictions to the ground truth trajectory. Note the increased uncertainty for the middle left part of Figure a) and top left part of Figure b).

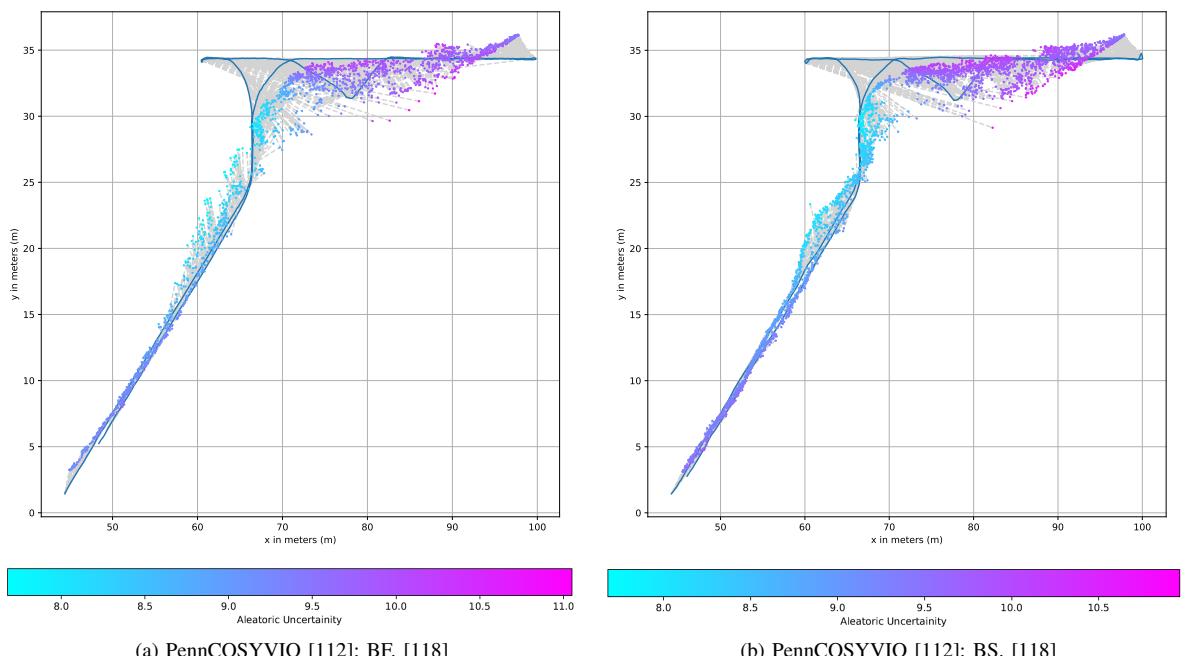
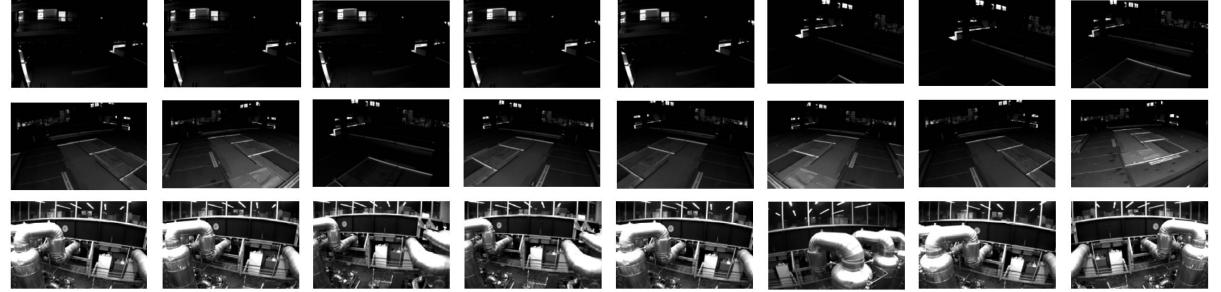


Fig. 39: Plot of the aleatoric uncertainty for the absolute pose prediction. Note the increased uncertainty for the top right part.

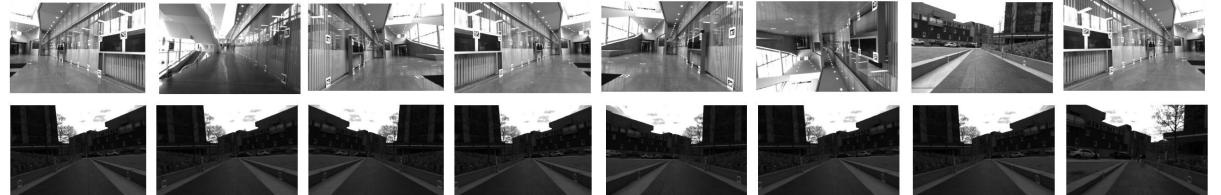


(a) EuRoC MAV [18]: MH-02-easy.

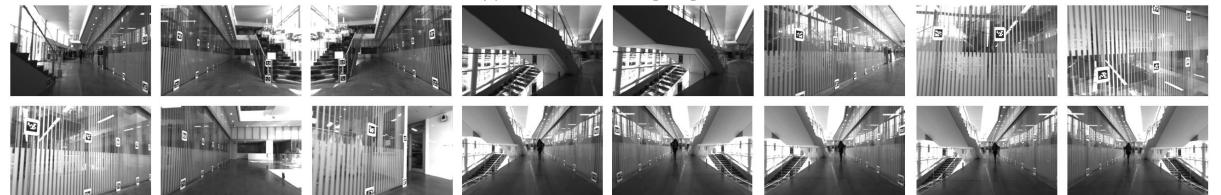


(b) EuRoC MAV [18]: MH-04-difficult.

Fig. 40: Exemplary images with a corresponding high uncertainty computed with the Bayesian model [71]. From [118].



(a) PennCOSYVIO [112]: BF.



(b) PennCOSYVIO [112]: BS.

Fig. 41: Exemplary images with a corresponding high uncertainty computed with the Bayesian model [71]. From [118].

11. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression

AUGUST 2023

29

TABLE IX: Evaluation results on the EuRoC MAV [18] dataset and comparison to state-of-the-art techniques. Absolute trajectory error (ATE) $e_{ATE,p}$ [131] calculated with [55]. **Bold** are best results.

At present, there is a plethora of visual odometry (VO) and inertial odometry (IO) techniques available, which primarily utilize the EuRoC MAV dataset for performance evaluation. Given the absence of a comprehensive survey comparing the results of all techniques, we present a summary of their results in Table IX and compare it with the results obtained from our RPR_I and RPR_V techniques using the absolute trajectory error (ATE) $e_{ATE,p}$ metric (see Section V). The EuRoC MAV dataset is cross-validated and evaluated on all testing sequences. The current state-of-the-art techniques employ alternative optimization strategies and exhibit improved performance on the EuRoC MAV dataset. ORB-SLAM [101] outperforms other methods on the MH-01, MH-02, MH-03, MH-04, V1-01, and V2-01 datasets, while ORB-SLAM2 [102] yields low ATE results on the MH-05, V1-03, and V2-02 datasets. The variance of results of various methods is very high. The utilization of a stereo camera has a significant impact, as evidenced by the significant improvement in the results of LSD-SLAM [40]. In addition, Qin et al. [116] improve the method that combines monocular images with IMU data by enhancing the method with stereo images. Our deep learning-based techniques are optimized based on the relative positional error, resulting in favorable performance as evaluated through the median relative metrics. However, it should be noted that the error increases when evaluated using the ATE metric. Comparing the results of IMUNet [36] for RPR_I and FlowNet [37] for RPR_V with the fusion techniques, wither FlowNet (MH-01 and V1-01), late fusion utilizing SSF [25] and BiLSTM layers (V1-03 and V2-02), and particularly MMTM [65] (on the remaining sequences) yield the lowest ATE error.

Method	MH-01	MH-02	MH-03	MH-04	MH-05	V1-01	V1-02	V1-03	V2-01	V2-02	V2-03
Hong et al. [54]	0.14	0.13	0.20	0.22	0.20	0.05	0.07	0.16	0.04	0.11	0.17
SVO + MSF [41]	0.14	0.20	0.48	1.38	0.51	0.40	0.63	-	0.20	0.37	-
OKVIS [78]	0.16	0.22	0.24	0.34	0.47	0.09	0.20	0.24	0.13	0.16	0.29
ROVIO [8]	0.21	0.25	0.25	0.49	0.52	0.10	0.10	0.14	0.12	0.14	0.14
VINS-monocular [115]	0.27	0.12	0.13	0.23	0.35	0.07	0.10	0.13	0.08	0.08	0.21
SVO [43]: monocular	0.17	0.27	0.43	1.36	0.51	0.20	0.47	-	0.30	0.47	-
SVO [43]: monocular, edge	0.17	0.27	0.42	1.00	0.60	0.22	0.35	-	0.26	0.40	-
SVO [43]: monocular, edge + prior	0.10	0.12	0.41	0.43	0.30	0.07	0.21	-	0.11	0.11	1.08
SVO [43]: monocular, BA	0.06	0.07	-	0.40	-	0.05	-	-	-	-	-
DSO [39]	0.05	0.05	0.18	2.50	0.11	0.12	0.11	0.93	0.04	0.13	0.16
VI-DSO [138]	0.041	0.041	0.116	0.129	0.106	0.057	0.066	0.095	0.031	0.060	0.173
SVO [43]: stereo	0.08	0.08	0.29	2.67	0.43	0.05	0.09	0.36	0.09	0.52	-
ORB-SLAM [101] (no LC)	0.03	0.02	0.02	0.20	0.19	0.04	-	-	0.02	0.07	-
ORB-SLAM2 [102]	0.035	0.018	0.028	0.119	0.060	0.035	0.020	0.048	0.037	0.035	-
LSD-SLAM [40]: monocular, no LC	0.18	0.56	2.69	2.13	0.85	1.24	1.11	-	-	-	-
LSD-SLAM [40]: stereo	-	-	-	-	0.066	0.074	0.089	-	-	-	-
Qin et al. [116]: stereo	0.54	0.46	0.33	0.78	0.50	0.55	0.23	-	0.23	0.20	-
Qin et al. [116]: monocular+IMU	0.24	0.18	0.23	0.39	0.19	0.10	0.10	0.11	0.12	0.10	0.27
Qin et al. [116]: stereo+IMU	0.18	0.09	0.17	0.21	0.25	0.06	0.09	0.18	0.06	0.11	0.26
Qin et al. [114]	0.120	0.120	0.130	0.180	0.210	0.068	0.084	0.190	0.081	0.160	-
Mu et al. [98]	0.551	0.402	0.997	0.704	0.672	0.521	0.652	-	0.232	0.617	-
Feng et al. [42]	0.111	0.074	0.173	0.143	0.205	0.077	0.143	0.093	0.082	0.100	0.233
IMUNet [36]	1.7606	2.8277	2.9113	3.7231	3.2198	1.4714	1.7456	1.4709	1.7779	1.7878	1.8935
FlowNet [37]	2.0312	2.1546	3.5234	3.3650	2.9856	1.7920	1.8673	1.7761	2.0221	2.1203	2.1952
Late Fusion (concat)	2.2651	1.9991	3.3241	3.5760	2.8914	1.7451	1.7818	1.3156	1.8011	1.7992	1.9185
Late Fusion (concat) + BiLSTM	2.1820	1.7792	2.2406	2.9761	2.3476	1.6782	1.7009	1.2810	1.7429	1.7224	1.8208
Late Fusion (SSF) [25]	2.2827	1.9621	3.5606	3.5281	3.0364	1.6774	1.7758	1.3459	1.7758	1.8828	1.9256
Late Fusion (SSF) [25] + BiLSTM	2.0110	1.8171	2.5411	3.1201	2.5431	1.5901	1.6906	1.2134	1.7123	1.7601	1.8567
MMTM [65] (3 modules)	1.8740	1.7541	1.9451	2.6712	2.1182	1.6002	1.6321	1.2998	1.7001	1.7866	1.8012

Chapter 12

Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

Contributing Article

Felix Ott, Lucas Heublein, David Rügamer, Bernd Bischl, and Christopher Mutschler. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments. Submitted to *Computer Vision and Image Understanding (CVIU)*, Elsevier, Available at *arXiv preprint arXiv:2304.07250v2 [cs.CV]*, July 2023.

Publisher Website

Paper available at: <https://arxiv.org/abs/2304.07250>

Copyright License

This work is licensed under a Creative Commons Attribution International 4.0 License (<https://creativecommons.org/licenses/by/4.0/>). © Copyright held by the owner/author(s).

Author Contributions¹⁵

Felix Ott is the corresponding author of this paper and contributed by the conceptualization of the paper (i.e., ideas and aims), methodology, software, validation, formal analysis, investigation, data curation (i.e., data recording, data management, maintaining research data, and maintaining the website), writing of the original draft, reviewing and editing,

visualization, and project administration. Lucas Heublein contributed for the methodology (i.e., creation of models), software, validation, investigation, and data curation (i.e., data management). David Rügamer contributed with the formal analysis, writing (i.e., review and editing), visualization (i.e., presentation of the published work), and supervision. Bernd Bischl contributed by supervision. Christopher Mutschler contributed with computing resources, writing (i.e., review and editing), supervision, and funding acquisition of the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

Datasets

Along this paper, the Industry scenario #4 datasets were published and are available at:
https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets

Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments[★]

Felix Ott^{a,b,d,*}, Lucas Heublein^a, David Rügamer^{b,c,d}, Bernd Bischl^{b,d} and Christopher Mutschler^a

^aFraunhofer IIS, Fraunhofer Institute for Integrated Circuits IIS, Nordostpark 84, Nuremberg 90411, Germany

^bLMU Munich, Ludwigstraße 33, Munich 80539, Germany

^cTechnical University of Dortmund, August-Schmidt-Straße 1, Dortmund 44227, Germany

^dMunich Center for Machine Learning (MCML), Ludwigstraße 33, Munich 80539, Germany

ARTICLE INFO

Keywords:

visual self-localization
structure from motion
pose regression
optical flow
pose graph optimization
recurrent pose fusion
synthetic transfer learning
challenging environment

Datasets and Source Code:

https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets

ABSTRACT

The localization of objects is a crucial task in various applications such as robotics, virtual and augmented reality, and the transportation of goods in warehouses. Recent advances in deep learning have enabled the localization using monocular visual cameras. While structure from motion (SfM) predicts the absolute pose from a point cloud, absolute pose regression (APR) methods learn a semantic understanding of the environment through neural networks. However, both fields face challenges caused by the environment such as motion blur, lighting changes, repetitive patterns, and feature-less structures. This study aims to address these challenges by incorporating additional information and regularizing the absolute pose using relative pose regression (RPR) methods. RPR methods suffer under different challenges, i.e., motion blur. The optical flow between consecutive images is computed using the Lucas-Kanade algorithm, and the relative pose is predicted using an auxiliary small recurrent convolutional network. The fusion of absolute and relative poses is a complex task due to the mismatch between the global and local coordinate systems. State-of-the-art methods fusing absolute and relative poses use pose graph optimization (PGO) to regularize the absolute pose predictions using relative poses. In this work, we propose recurrent fusion networks to optimally align absolute and relative pose predictions to improve the absolute pose prediction. We evaluate eight different recurrent units and construct a simulation environment to pre-train the APR and RPR networks for better generalized training. Additionally, we record a large database of different scenarios in a challenging large-scale indoor environment that mimics a warehouse with transportation robots. We conduct hyperparameter searches and experiments to show the effectiveness of our recurrent fusion method compared to PGO.

1. Introduction

For various path planning applications, such as robotic systems operating in vast warehouses, obtaining an accurate localization of objects is crucial (Radwan et al., 2018; Löffler et al., 2018). To achieve this, the robot's pose recognition, which includes its position and orientation, must be highly precise. While some localization systems use LiDAR, radio, or radar-based systems (Stahlke et al., 2022) or inertial sensors (do Monte Lima et al., 2019), visual self-localization

using monocular cameras has gained popularity with the advancement of deep learning techniques (Kendall et al., 2015). The effectiveness of pose estimation techniques depends heavily on the suitable invariance properties of the features available (Ott et al., 2022a).

The 3D structure of a scene can be estimated from a sequence of 2D images using structure from motion (SfM) methods (Venkataraman, 2020; Resch et al., 2015; Jiang et al., 2020; Brachmann et al., 2023; Yu et al., 2023; Li et al., 2023b), which involve camera motion estimation by detecting and matching feature points between pairs of images, and 3D structure estimation by triangulating the feature points. Absolute pose regression (APR) techniques (Kendall et al., 2015; Löffler et al., 2018; Radwan et al., 2018) use neural networks to extract features from images and estimate the 6DoF global pose of an object with respect to a known coordinate system. Classical visual odometry (VO) methods (Mansur et al., 2017) estimate camera motion by analyzing consecutive images to compute the relative pose of an object. Relative pose regression (RPR) methods (Wang et al., 2017; Iyer et al., 2018; Kreuzig et al., 2019; Idan et al., 2023) have emerged, which predict the relative pose by using convolutional neural networks (CNNs) and recurrent neural networks (RNNs). RPR methods based on optical flow, which captures the motion of pixels between two frames, are particularly promising due to their robustness to environmental changes, as evidenced by recent studies

[★] Evaluating the performance of absolute pose regression methods, i.e., SfM and PoseNet, in challenging scenarios such as repetitive patterns, motion blur, structure-less surfaces, and difficult lighting conditions. Fusing absolute and relative poses using either PGO or recurrent networks.

*Corresponding author

✉ felix.ott@iis.fraunhofer.de (F. Ott); heublein@iis.fraunhofer.de (L. Heublein); david.ruegamer@stat.uni-muenchen.de (D. Rügamer); bernd.bischl@stat.uni-muenchen.de (B. Bischl); christopher.mutschler@iis.fraunhofer.de (C. Mutschler)

✉ <https://www.slds.stat.uni-muenchen.de/people/ott/> (F. Ott); <https://www.slds.stat.uni-muenchen.de/people/ruegamer/> (D. Rügamer); <https://www.statistik.uni-muenchen.de/personen/professoren/bischl/index.html> (B. Bischl); <https://cmutschler.de/> (C. Mutschler)

ORCID(s): 0000-0002-4392-0830 (F. Ott); 0000-0001-6670-3698 (L. Heublein); 0000-0002-8772-9202 (D. Rügamer); 0000-0001-6002-6980 (B. Bischl); 0000-0001-8108-0230 (C. Mutschler)

[in](https://www.linkedin.com/profile/view?id=felix-ott-494b06146) <https://www.linkedin.com/profile/view?id=felix-ott-494b06146> (F. Ott), <https://www.linkedin.com/profile/view?id=david-ruegamer> (D. Rügamer), <https://www.linkedin.com/profile/view?id=christopher-mutschler-28431576> (C. Mutschler)

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

332

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

(Muller and Savakis, 2017; Muller et al., 2016; Zhou et al., 2020; Ott et al., 2020).

Each method for absolute pose prediction, such as SfM and APR, and RPR, has its own drawback and each pose has different sources of noise, as highlighted by Sattler et al. (2019). To improve the performance of these methods, it is desirable to combine absolute poses (from SfM or APR) and relative poses (through RPR from optical flow). While some approaches combine both fields or different modalities by a common representation between networks (Ott et al., 2022a; Briege et al., 2022), others directly combine the absolute and relative poses (Mitsuki et al., 2021). However, fusing absolute and relative poses is a challenging task since RPR requires knowledge of the global pose, which may contain errors. Pose graph optimization (PGO) (Brahmbhatt et al., 2018; Mirowski et al., 2018) is a technique that estimates smooth and globally consistent pose predictions during inference. Another approach is to use a fusion module based on recurrent cells (Ott et al., 2020), which can learn an improved (i.e., smoothed), long-term trajectory, but requires time-distributions (i.e., applies a layer to every temporal slice of an input) to learn the orientation of the relative to the global pose. Despite recent advances, optimizing the absolute pose with neural networks remains an open research topic.

Another research goal of visual self-localization is to develop methods that can adapt to new and unknown scenes or remain robust to changes in the environment (Idan et al., 2023; Winkelbauer et al., 2021; Wang and Qi, 2023; Acharya et al., 2023), such as adding or removing racks in warehouses. To achieve this, some methods learn multi-scene APR (e.g., using Transformers) to learn multiple scenes in parallel (Shavit et al., 2022), utilize auto-encoders (Shavit and Keller, 2022), or employ transfer learning between scenes (Chidlovskii and Sadek, 2021). To improve the initial weights, we pre-train the APR and RPR models using synthetic data.

Contributions. The primary objective of this work is to optimize absolute poses by fusing absolute and relative poses. To achieve this goal, the following steps are taken: (1) We conduct a large hyperparameter search for SfM to retrieve absolute poses. PoseNet (Kendall et al., 2015) is alternatively evaluated as an APR technique. (2) The relative poses are regressed from optical flow computed with the Lucas-Kanade (Baker and Matthews, 2004) algorithm. (3) We evaluate PGO (Mirowski et al., 2018) to optimize the pose, and conduct a benchmark of eight recurrent and 17 convolutional, recurrent, and Transformer networks for absolute and relative pose fusion. (4) As there is currently no publicly available dataset to evaluate large-scale challenging environments, we record and publish¹ a large database of various scenes and changes between scenes. This dataset is used to conduct experiments on the robustness of methods against volatile environments. (5) We develop a simulation framework to generate and pre-train the APR and RPR

models on synthetic data. Advanced techniques may be used as black box models in place of the baseline models (SfM, APR, RPR) to further enhance the results. However, the focus of this study is on assessing the performance of the fusion techniques with respect to environmental changes and motion dynamics on the localization task.

The remainder of this article is organized as follows. We discuss related work in Section 2. Section 3 introduces our method, including SfM, APR, RPR, and fusion modules. We present novel datasets and our simulation for data generation in Section 4, and discuss experimental results in Section 5.

2. Related Work

The field of SfM and APR has seen an extensive number of publications in recent years, making it a broad and diverse research area. However, the main focus of this work is on the fusion of absolute and relative poses. For a comprehensive overview of SfM approaches, please refer to Jiang et al. (2020); Zheng et al. (2018); Piasco et al. (2018); Brachmann et al. (2023); Radanovic et al. (2023), while for an overview of APR techniques, please see Sattler et al. (2019); Ott et al. (2022a); Xu et al. (2022); Blanton (2021); Qiao et al. (2023). See Boittiaux et al. (2022); Pepe and Lasenby (2023), for an overview of APR loss functions. In Section 2.1, we introduce techniques for RPR estimation using optical flow. State-of-the-art techniques for absolute and relative pose fusion are discussed in Section 2.2. In Section 2.3, we present related work for scene generalization.

2.1. RPR from Optical Flow

Numerous techniques exist for estimating the relative pose from successive image pairs by utilizing optical flow, including Flowdometry (Muller and Savakis, 2017; Muller et al., 2016), ViPR (Ott et al., 2020), DeepVIO (Han et al., 2019), KFNet (Zhou et al., 2020), and the model by Ott et al. (2022a). These methods employ either FlowNet or FlowNet-Simple by Dosovitskiy et al. (2015) or FlowNet2 (Ilg et al., 2017) for estimating optical flow. LS-VO (Costante and Ciarfuglia, 2018) uses an autoencoder to predict optical flow. In contrast, Zhi-Yu et al. (2021) utilizes the relative pose as auxiliary information for optical flow prediction of FlowNet2. Our method uses the Lucas-Kanade (Baker and Matthews, 2004) algorithm. DeepVO (Wang et al., 2017), CTCNet (Iyer et al., 2018), DistanceNet (Kreuzig et al., 2019), and MotionNet (Ding et al., 2020) do not rely on optical flow but combine CNNs with RNNs, such as stacked LSTMs, bidirectional LSTMs, or fully connected layers, to model sequential dynamics and relationships to predict relative poses.

2.2. APR & RPR Fusion

LM-Reloc (von Stumberg et al., 2020) formulates its loss based on the Levenberg-Marquardt algorithm to improve direct image alignment through learned features. It also incorporates an RPR model to bootstrap the direct image alignment. ViPR (Ott et al., 2020) enhances absolute poses by concatenating relative poses predicted from optical

¹Datasets and source code: https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets

flow and absolute poses, and then refining them using an LSTM network. MapNet+PGO (Brahmbhatt et al., 2018) refines predicted poses from APR and RPR using PGO. VLocNet (Valada et al., 2018) estimates global poses and combines it with VO, while VLocNet++ (Radwan et al., 2018) adds features from semantic segmentation. RCNN (Lin et al., 2019) fuses relative and global sub-networks to smooth VO and prevent drift. Mitsuki et al. (2021) propose a graph neural network that uses image-to-nodes and image-to-edges to create similarity-preserving mappings, with nodes representing absolute features and edges representing relative features. Kalman filters are frequently employed for pose fusion, such as in the work by Emter et al. (2019). This approach combines absolute and relative measurements, leading to a correlation between a past state and present state that creates correlations. Our objective is to mitigate these correlations by utilizing an optimal recurrent cell. In general, LSTM cells are employed for representing temporal dependencies in localization tasks. Recently, Transformer networks have been applied in visual self-localization, as demonstrated by the methods proposed by Shavit et al. (2022); Li and Ling (2022); Kim and Kim (2023); Li et al. (2023a). Nevertheless, Transformers are not adequate for integrating absolute and relative poses with their low-dimensional information. Also, the bidirectional properties (Graves et al., 2009) of LSTMs are unfeasible for pose modeling. Ruan et al. (2023) fuse APR with scene coordinate regression. While Lu and Lu (2019) combine a depth map from a single-view depth network with relative poses from a pose network, Yang et al. (2020) proposed D3VO that combines the depth map with relative poses from PoseNet. In contrast, Zhan et al. (2020) proposed combination of a depth network with optical flow. However, this method only performs visual odometry.

Das and Dubbelman (2018) fuse an absolute pose (from GNSS data) with a relative pose (from vehicle odometry). The following two works integrate relative poses from inertial data with absolute poses from visual data. VINet (Clark et al., 2017) utilizes a concatenation approach to merge relative features extracted by an inertial LSTM encoder with absolute features extracted by a visual encoder. On the other hand, VI-DSO (von Stumberg et al., 2018) employs a combined energy functional to jointly estimate camera poses and sparse scene geometry by minimizing the visual and inertial measurement errors.

2.3. Simulation for Scene Generalization

Todays regression-based techniques for APR are scene-specific with respect to their training and evaluation. They rely on the coordinate system of the training dataset and exhibit poor generalization across different scenes (Chidlovskii and Sadek, 2021). The Transformer network, proposed by Shavit et al. (2022), learns multi-scene APR. The model employs encoders to aggregate activation maps via self-attention and decoders to transform latent features and scenes encoding into predicted pose. This allows the model to learn informative features while embedding multiple

scenes in parallel. To address the issue of dataset shift, Chidlovskii and Sadek (2021) developed a deep adaptation network that can learn scene-invariant image representations for transfer learning. They use adversarial learning to generate such representations for the model transfer. Wang and Qi (2023) report that single-input images can cause confusion in relocalization when dealing with scenes that share similar views but differ in position, which motivates the use of a time-distributed network. Furthermore, they highlight the difficulty of relocalization in variable or dynamic scenes. Wang and Qi (2023) created a variable scene dataset comprising three scenes: an office, a bedroom, and a sitting room. They use semi-automatic processing to develop their MMLNet method, which can regress both camera pose and scene point cloud. Similarly, our approach involves fusing point clouds and relative poses. Acharya et al. (2023) uses a generative network to model fake synthetic images for APR.

While APR techniques learn absolute scene parameters, RPR methods can localize in unseen environments by learning only the residual pose between image pairs. However, their performance is notably reduced in unfamiliar scenes. To enhance the generalization of RPR methods, Idan et al. (2023) improved their performance by aggregating paired feature maps into latent codes. Meanwhile, Winkelbauer et al. (2021) proposed an RPR approach (based on ResNet) that introduces changes to the model architecture, such as an extended regression part, hierarchical correlation layers, and uncertainty and scale information, to better generalize to new scenes.

In summary, prior works have focused on introducing auxiliary information to the model to learn general and scene-specific features that can generalize to various scenes. However, no existing approach learns scene information from simulated environments (absolute) or motion from simulated dynamics (relative).

3. Methodology

Firstly, we present to reconstruct a point cloud from an image dataset using SfM (see Section 3.1). Section 3.2 describes an alternative approach based on APR. The Lucas-Kanade method for computing optical flow and our RPR network are presented in Section 3.3. Absolute and relative pose fusion are described based on these APR and RPR methods. Additionally, we evaluate PGO in Section 3.4 and propose a framework of models that utilize various RNNs in Section 3.5. Finally, in Section 3.6, the rationale for pre-training APR and RPR to improve generalizability to various scenarios is discussed. Figure 1 presents a method overview of all steps.

3.1. Structure from Motion

This section outlines the steps involved in using SfM to create a point cloud. The SfM pipeline is proposed in Figure 2, while Table 1 provides a summary of the hyperparameters used in each step.

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

334

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

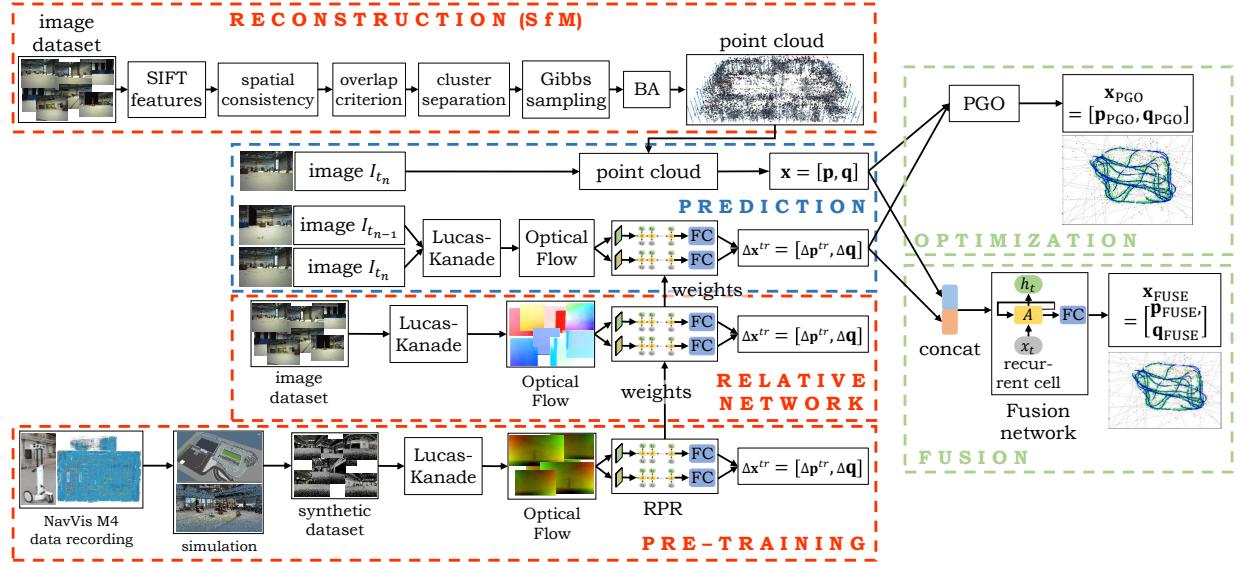


Figure 1: Method overview. First, a point cloud is constructed from an image dataset using SfM to extract features, spatial consistency, overlap criterion, cluster sampling, and bundle adjustment (BA). Second, we train a small convolutional recurrent neural network to predict the relative pose Δx between two consecutive images. For evaluation, the absolute pose x from the point cloud and the relative pose Δx^{tr} from the RPR model is retrieved for a query image. Last the absolute pose is optimized with either PGO or a recurrent network.

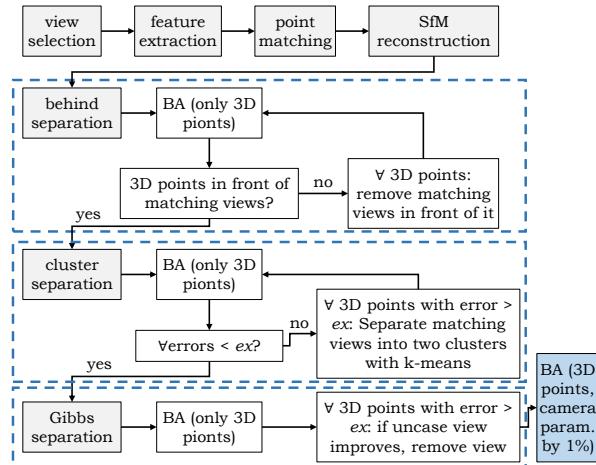


Figure 2: SfM pipeline using bundle adjustment (BA) to reconstruct a point cloud from input images.

SfM. We adopt the source code provided by Venkataraman (2020) as the baseline for our SfM pipeline, which is used to recover 3D structure of a given environment. SfM is employed to determine the relative pose of each image with respect to the first image. Inspired by the findings of Resch et al. (2015) that suggest the use of every fifth image for reconstruction, we manually pre-select around 600 images from each dataset (comprising between 7,000 to 138,000 images) for point cloud reconstruction. Additionally, we obtain the intrinsic matrix, which is essential for recalibration, by capturing images of a checkerboard using our camera

Table 1
Overview of SfM hyperparameters.

Parameter	Description
<i>sc</i>	Spatial consistency of the point neighborhood in pixel
<i>oc</i>	Overlap criterion of floor pixels between images in %
<i>mm</i>	Minimal number of matches for each point over all images
<i>ex</i>	Exclude points with reprojection error larger than separation limit in pixel
<i>gibbs</i>	Boolean of <i>gibbs</i> factor of pixel improvement by excluding single points
<i>std</i>	Standard deviation for point exclusion for BA

setup. For the sake of reproducibility, we report the following calibration matrix:

$$K = \begin{bmatrix} 548.44934818 & 0.0 & 317.73762648 \\ 0.0 & 540.17600512 & 249.00614224 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} \quad (1)$$

We ensure to avoid matching two points from the same image with a single point from the point cloud.

Feature Extraction. We utilize the cv2 library (OpenCV, 2022) to extract image features, which yields descriptors of size 128 for each image. In a pre-defined study, we asses the impact of three feature extraction techniques, namely, scale-invariant feature transform (SIFT) proposed by Lowe

(2004), speeded up robust features (SURF) introduced by Bay et al. (2006), and oriented fast and rotated brief (ORB) proposed by Rublee et al. (2011), on the reconstruction performance. Based on our evaluation, we select SIFT as the feature extraction technique for our pipeline.

Spatial Consistency & Overlap Criterion. In addition, we employ spatial consistency to enhance the discriminative capability of the raw feature points by evaluating the matching quality of feature points in a larger spatial neighborhood. We follow the approach proposed in Jiang et al. (2020), which involves computing the ratios of features that fall into two corresponding regions. We introduce a hyperparameter sc that specifies the spatial consistency of the point neighborhood in terms of the number of pixels. We also apply an overlap criterion to filter out unnecessary image pairs. We divide the ground floor into a grid and require each image to contain a specific percentage (controlled by the parameter oc) from these grid points.

Cluster Separation. Points from multiple images can correspond to the same point in the point cloud. To handle such cases, we partition these image points into cluster and explore the possibility of improving the match separation, as proposed in Jiang et al. (2020). This results in the creation of additional clusters in the point cloud. We use the hyperparameter mm to determine the number of matches. For neighborhood clustering, we use k-means with two clusters. We apply this strategy iteratively with bundle adjustment (BA) until the hyperparameter ex falls below a predefined threshold in term of pixels.

Gibbs Sampling. We first apply BA. Next, we iterate through all image points for each point of the point cloud. We employ Gibbs sampling to iteratively remove one point from the selected image points and evaluated whether the remaining points improve the performance of the BA. We adjust the hyperparameter $gibbs$ for this process.

Bundle Adjustment. We adopt the sparse BA method proposed by Github (jahdiel) (2016). Additionally, we define parameters to regularize the point cloud to remain within the environment. In all BA steps, we keep the rotation matrix fixed, and only change the rotation matrix of the last BA step. We tune the hyperparameter std , which represents the standard deviation for point exclusions.

3.2. Absolute Pose Regression

In order to reduce the computational requirements associated with hyperparameter tuning for SfM (that we provide in Section 5.1), an alternative approach involves using APR methods, as discussed in Section 2.2. Rather than tuning hyperparameters, a CNN-based APR model can learn to directly regress the camera pose from a single input image in conjunction with their corresponding ground truth poses. In our case, we utilize a time-distributed network that takes a set of three consecutive images as input (at timesteps t_{n-2} , t_{n-1} , and t_n) to predict absolute positions $\mathbf{p} \in \mathbb{R}^3$ in Euclidean

coordinates and absolute orientations $\mathbf{q} \in \mathbb{R}^4$ as quaternions of the absolute pose $\mathbf{x} = [\mathbf{p}, \mathbf{q}]$. We use the PoseNet architecture (Kendall et al., 2015) based on GoogLeNet (Szegedy et al., 2015) with time-distribution (Ott et al., 2020). Our network includes a fully connected (FC) layer of 2,048 units, and two parallel FC layers, each with three and four units. We minimize the root mean squared error (RMSE) loss function

$$\mathcal{L}_{\text{APR}} = \|\hat{\mathbf{p}} - \mathbf{p}\|_2^2 + \beta_1 \left\| \hat{\mathbf{q}} - \frac{\mathbf{q}}{\|\mathbf{q}\|_2} \right\|_2^2, \quad (2)$$

between the predicted pose $\mathbf{x} = [\hat{\mathbf{p}}, \hat{\mathbf{q}}]$ and ground truth pose $\hat{\mathbf{x}} = [\hat{\mathbf{p}}, \hat{\mathbf{q}}]$, weighted by the hyperparameter $\beta_1 = 50$. We use a batch size of 50, the Adam optimizer without decay, and a learning rate of 10^{-4} .

3.3. Relative Pose Regression from Optical Flow

We utilize the Lucas-Kanade (Baker and Matthews, 2004) algorithm to compute the optical flow between two images captured at timesteps t_{n-1} and t_n for learning the relative movements and rotations of an object. The optical flow is a vector field representation of the movement of objects or surfaces in a sequence of images or video, as captured by a camera. Each vector in the optical flow field corresponds to the displacement of a small portion of the image from frame t_{n-1} to frame t_n . Hence, we receive the pixel movements in u and v direction. The Lucas-Kanade algorithm is a classic differential method that assumes uniform motion of objects in a small local neighborhood and solves a system of linear equations relating the spatial and temporal derivatives of the image intensity to the local motion parameters. We present an exemplary visualization of optical flow in Figure 3.

We propose the following recurrent network for the purpose of regressing the relative pose from optical flow. To achieve this, we initially subject the vector input, which has a

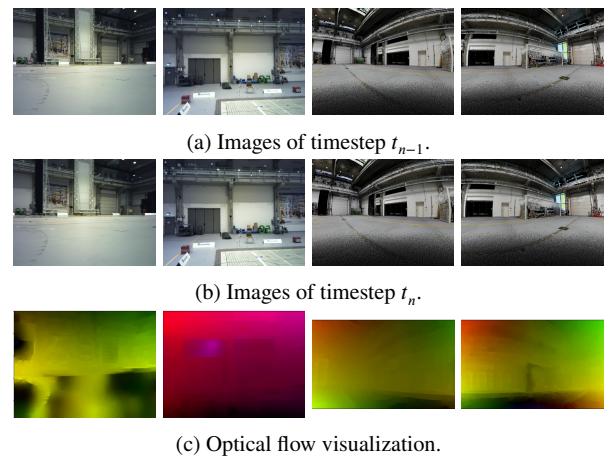


Figure 3: Exemplary optical flow visualizations (c) between two consecutive images (a and b) of real-world scenarios (image 1 and 2) and simulated scenarios (image 3 and 4). Note the small movement between timestep t_{n-1} (a) and timestep t_n (b).

size of $480 \times 640 \times 2$ pixels, to 4×4 mean average pooling with stride 1, resulting in a tensor size of $120 \times 160 \times 2$. In this case, u and v represent the third dimension of the optical flow. We then proceed to search for an optimal network configuration, which may involve batch normalization, adding one or two stacked LSTM layer, and ReLU, softmax, or no activation. Our chosen network configuration involves the absence of batch normalization and activation, and the use of one LSTM layer with 50 units for each direction of u and v by slicing the third dimension of the optical flow. The output of the LSTM layers is then concatenated, and two FC layers of size 3 and 4 are added to predict the relative pose $\Delta x = [\Delta p^{tr}, \Delta q]$, i.e., the relative position (translation) $\Delta p^{tr} \in \mathbb{R}^3$ and the relative orientation (rotation) $\Delta q \in \mathbb{R}^4$. To minimize error, we apply a mean squared error (MSE) loss function

$$\mathcal{L}_{RPR} = \|\Delta \hat{p}^{tr} - \Delta p^{tr}\|_2 + \beta_2 \left\| \Delta \hat{q} - \frac{\Delta q}{\|\Delta q\|_2} \right\|_2, \quad (3)$$

with the ground truth relative pose $\Delta \hat{x} = [\Delta \hat{p}^{tr}, \Delta \hat{q}]$. Therefore, we transform the global coordinate systems of consecutive cameras to a local coordinate system $\bar{p} = Rp$ with the rotation matrix R , such that $R^T = R^{-1}$ and $R^T R = RR^T = I$. The relative poses Δp^{tr} are then the differences between the rotated poses. We weight the orientational loss function with $\beta_2 = 50$. Additionally, we use a batch size of 50, the Adam optimizer without decay, and a learning rate of 10^{-4} . Finally, the relative ground truth labels are transformed based on the current orientation of the absolute pose.

3.4. Pose Graph Optimization

We employ the PGO algorithm (Mirowski et al., 2018) as the state-of-the-art method for fusing absolute and relative poses, which provides smooth and globally consistent pose estimates. PGO can be formulated as a non-convex minimization problem represented by a graph with vertices corresponding to the estimated global poses and edges representing the relative poses. The goal is to ensure that the refined relative poses align with the input relative poses (Brahmbhatt et al., 2018; Ott et al., 2022a). To reduce run time complexity, we split the absolute and relative poses into 100-sample chunks with a 20-sample overlap. We conducted a search for the optimal parameters that results in the lowest positioning error.

3.5. APR-RPR Fusion based on Recurrent Cells

Our contribution is a recurrent model for fusing absolute and relative poses to achieve a smoother absolute pose estimation and optimize localization error. To accomplish this, we concatenate the output pose of the absolute method (refer to Section 3.1 and Section 3.2) of size $(N_t \times BS \times 7)$ and the output pose of the relative method (refer to Section 3.3) of size $(N_t \times BS \times 7)$, resulting in a network input of size $(N_t \times BS \times 14)$, where N_t is the number of timesteps and BS is the batch size. The objective is to predict an improved absolute pose at timestep t_n from the concatenated absolute and relative poses of timesteps $\{t_{n-N_t-1}, \dots, t_n\}$. For pose

refinement, we implement the following recurrent network, which includes one or two stacked RNN cells. The output of the first cell has a size of $(N_t \times BS \times 14)$, while the output of the second cell is of size $(BS \times r_u)$, where r_u represents the number of units of the RNN cell. Finally, we include two FC layers, similar to the APR model (refer to Section 3.2), with the RMSE loss function

$$\mathcal{L}_{APR-RPR} = \|\hat{p} - p\|_2^2 + \beta_3 \left\| \hat{q} - \frac{q}{\|q\|_2} \right\|_2^2, \quad (4)$$

and weighting $\beta_3 = 50$, a batch size of 100, the Adam optimizer without decay, and a learning rate of 10^{-4} . In Section 5.4, we present a hyperparameter search for N_t , r_u , and the number of stacked recurrent cells (one or two).

The vanishing gradient problem is a major issue with RNNs, which occurs when gradients become too small as they propagate backwards through time during weight updates. This can be problematic in our fusion model due to small orientation entries (i.e., quaternions are below 1) and hinders the learning of long-term dependencies. To address this issue, we explore various recurrent cells that can allow the model to learn the dynamics of an object (i.e., slow moving robot versus fast moving human) on the low-level pose representation. Object movements typically follow predictable, non-chaotic patterns that follow physical behavior (Ott et al., 2022b). One solution is to use gating mechanisms such as those found in long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (GRU) (Chung et al., 2014). Another option is the minimal gated unit (MGU) (Zhou et al., 2016; Heck and Salem, 2017), which is a simplified version of LSTM and GRU that uses only an update and reset gate. The recurrent additive network (RAN) (Lee et al., 2017) is another type of gates RNN that uses purely additive latent state updates. For greater parallelism, we evaluate the simple recurrent unit (SRU) (Lei et al., 2017), which simplifies computations as the majority of computations for each step are independent of recurrence. Additionally, quasi-recurrent neural networks (QRNN) (Bradbury et al., 2017) apply minimalistic recurrent pooling functions in parallel across channels. Balduzzi and Ghifary (2017) proposed strongly-typed RNN (TRNN), which learns simple semantic interpretations via dynamic average pooling. The chaos free network (CFN), proposed by Laurent and von Brecht (2017), is a type of RNN that models non-chaotic dynamics.

Furthermore, we evaluate convolutional networks, such as FCN (Wang et al., 2016), TCN (Bai et al., 2018), ResNet (Wang et al., 2016), ResCNN (Zou et al., 2019), InceptionTime (Fawaz et al., 2020), XceptionTime (Rahimian et al., 2019), and OmniScaleCNN (Tang et al., 2022), a combination of recurrent with fully convolutional networks, i.e., LSTM-FCN (Karim et al., 2017), GRU-FCN (Elsayed et al., 2019), and MLSTM-FCN (Karim et al., 2019), Transformer models, i.e., TST (Zerveas et al., 2021), TSPerceiver (Jaegle et al., 2021), and TSSequencePlus (Tatsunmai and Taki, 2022), and mWDN (Wang et al., 2018), XCM (Fauvel et al., 2022), and gMLP (Liu et al., 2021).

Table 2

Overview of the visual Industry scenario #4 datasets recorded in a large-scale indoor environment with a robot or handheld.

Dataset	Setup	# Images
Train 1	Robot, clear environment	92,668
Train 2	Robot, four absorber walls with objects	19,872
Train 3	Robot, four absorber walls	138,233
Train 4	Handheld, person 1, four absorber walls	27,856
Train 5	Handheld, person 2, four absorber walls	28,113
Train 6	Robot, open environment with objects	114,998
Train 7	Robot, open environment with objects and labyrinth	29,240
Train 8	Robot, open environment with object, labyrinth, and absorber walls	110,923
Test 1	Robot, clear environment	23,168
Test 2	Robot, one absorber wall	30,379
Test 3	Robot, two absorber walls	24,752
Test 4	Robot, three absorber walls	26,877
Test 5	Robot, four absorber walls with objects	4,968
Test 6	Robot, four absorber walls	34,559
Test 7	Handheld, person 1, four absorber walls	6,964
Test 8	Handheld, person 2, four absorber walls	7,029
Test 9	Robot, open environment with objects	28,750
Test 10	Robot, open environment with objects and labyrinth	7,311
Test 11	Robot, open environment with objects, labyrinth, and absorber walls	27,732

3.6. Simulation-Augmented Pre-Training

Pre-training neural networks using simulated data can enhance their performance in various tasks, especially in the case of data scarcity, domain adaptation, data augmentation, and providing a good initialization for transfer learning to real data (Zhang et al., 2017; Shrivastava et al., 2017). Previous studies in the context of visual self-localization (Winkelbauer et al., 2021; Idan et al., 2023; Wang and Qi, 2023) have demonstrated the potential benefits of pre-training on synthetic data for improved performance. To this end, we develop a simulation environment with a loaded point cloud of the real-world environment to generate a large number of samples and augment the APR and RPR models for better initial weights, improved adaptability to unseen scenarios, and enhanced transfer learning to changes in the environment. Figure 4 illustrates the data generation process. Initially, we record a large dataset using a NavVis M4 system (see Figure 4a) while moving in an environment with seven large black absorber walls and two warehouse racks, as shown in Figure 4b. Next, we create a detailed point cloud (see Figure 4c) from this dataset, which was then loaded into the simulation framework implemented by Shhuna GmbH (see Figure 4d). We only use the large hall of the simulation framework (left part) for data generation. Using a fish-eye camera (in the simulation), we generated 319,955 images mimicking a slow moving robot, see Figure 3 (third and fourth column) for exemplary images. We concurrently capture images from three cameras, including one front-facing camera and two side-facing cameras. We utilized this dataset to pre-train the APR and RPR models on a large dataset.

4. Experiments

In the following section, we provide details regarding the data collection and experiments. Specifically, Section 4.1 offers an overview of our datasets and the challenges associated with localization. Section 4.2 provides a summary of our experiments. Finally, in Section 4.3, we describe the hardware setup and evaluation metrics used in our study.

4.1. Datasets & Challenges

Numerous datasets are currently accessible for assessing APR techniques. Nevertheless, the present datasets possess limitations such as being captured outside the industrial environments or with equipment such as micro aerial vehicles (MAVs) or handheld devices, or are inadequate for

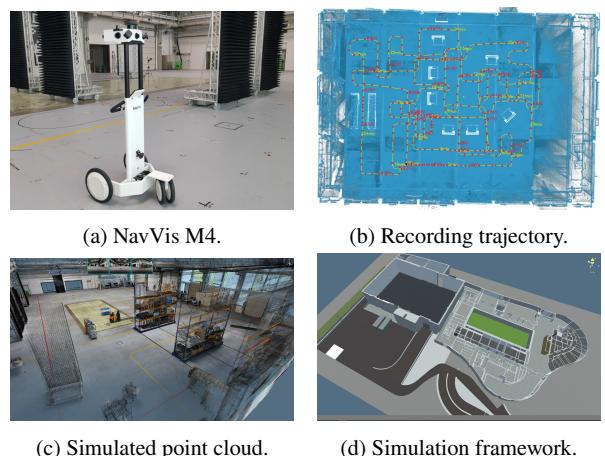


Figure 4: Creation of simulated images for pre-training.

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

338

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

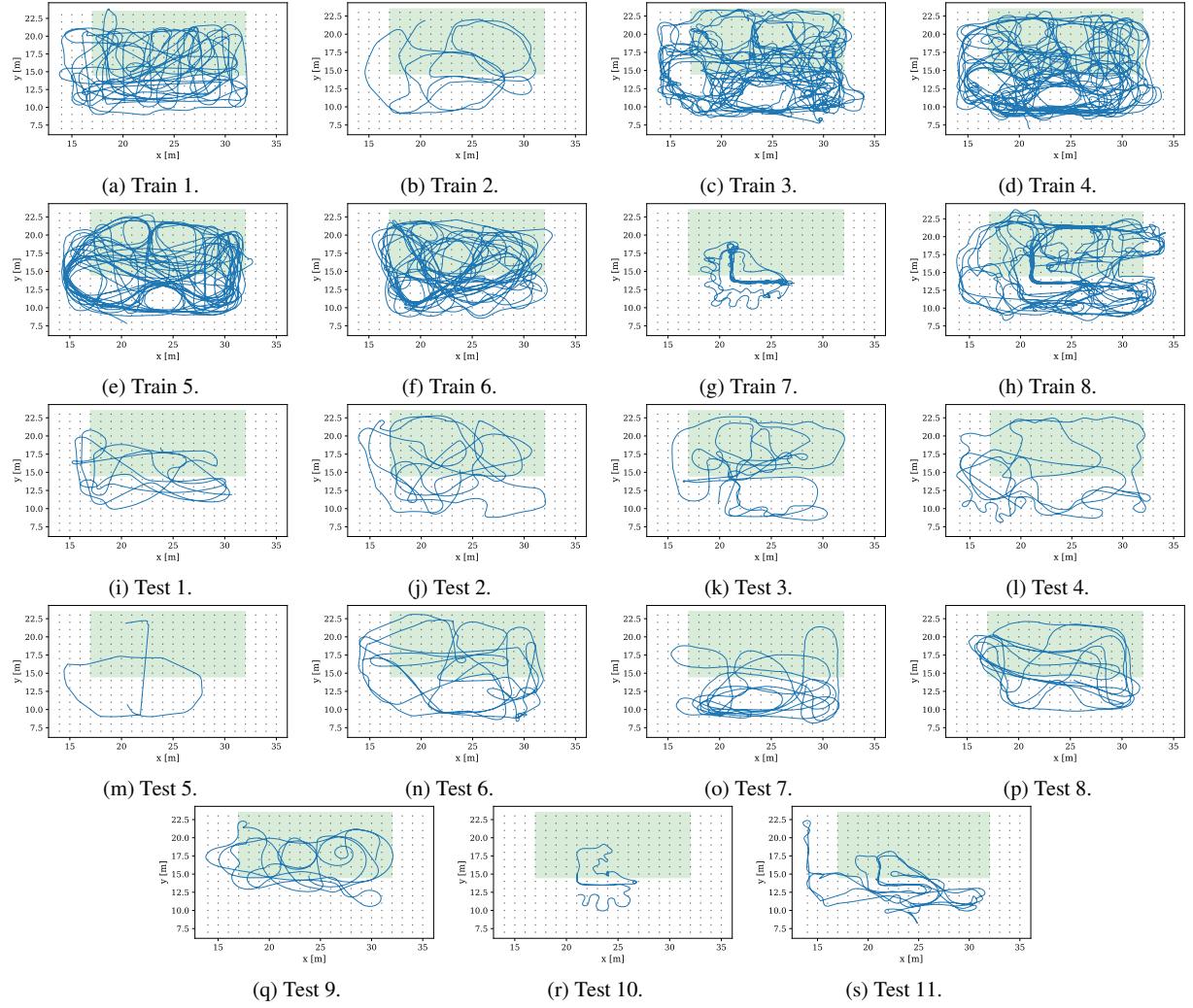


Figure 5: Trajectories of the Industry datasets with $x \in [14, 34]$ and $y \in [7, 23]$.

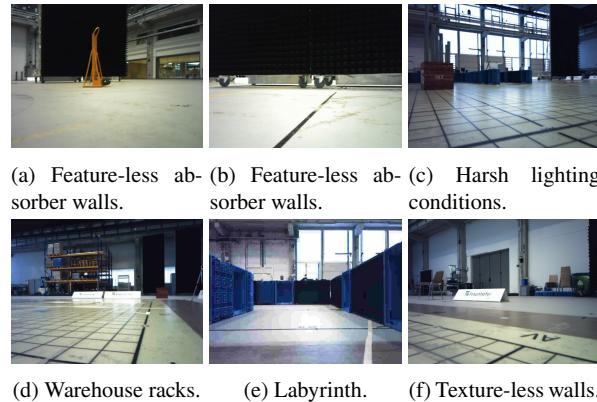
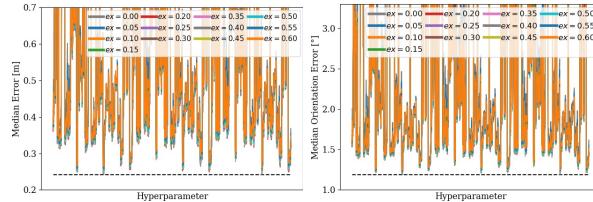
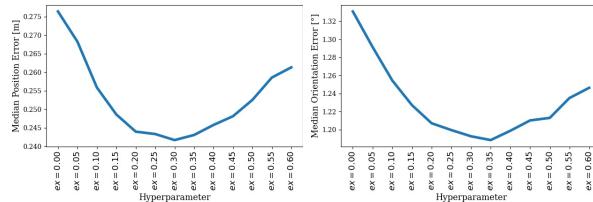


Figure 6: Exemplary challenging images of the large-scale indoor environment.

evaluating specific scenarios like changes in the environment. Therefore, we have captured the Industry dataset in

a vast large-scale industrial environment covering an area of $1,320\text{m}^2$, similar to the environment in Löffler et al. (2018); Ott et al. (2020, 2022a); Stahlke et al. (2022). A small robotic recording platform is constructed, equipped with an Orbbec3D camera featuring an RGB image resolution of 640×480 pixels and a recording frequency of 23 Hz. To measure reference poses at a high-precision level ($< 1\text{mm}$), a motion capture system operating at 140 Hz is utilized. Eight training and 11 testing datasets are recorded, comprising 10 distinct scenarios, as presented in Table 2. Trajectories are depicted in Figure 5. Changes are introduced to the environment between recordings to assess the model's robustness against volatile objects, i.e., the removal or addition of absorber walls, and the ability to generalize and adapt to various scenarios and motion dynamics. Initially, a large-scale clear environment without objects was recorded. Subsequently, we introduced one, two, three, or four absorber walls, along with smaller objects. Additionally, a small "labyrinth" in an L-like configuration was constructed,

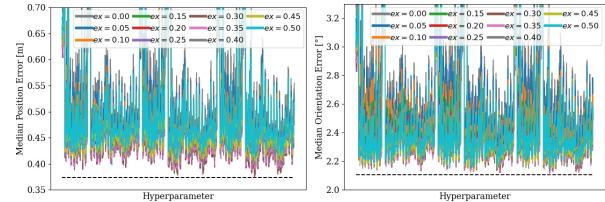
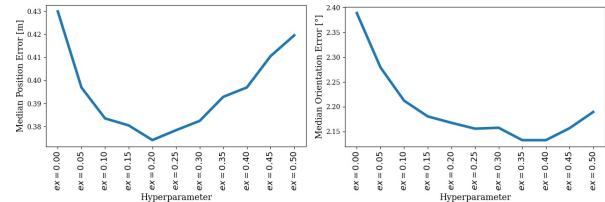
Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

(a) Median position error in m . (b) Median orientation error in $^{\circ}$.**Figure 7:** Evaluation of all SfM point clouds (for different hyperparameters, see x-axis) for the robot train 3 and test 6 datasets. The dashed lines indicate the lowest error.(a) Median position error in m . (b) Median orientation error in $^{\circ}$.**Figure 8:** Evaluation of the best SfM point cloud (from Figure 7) for different hyperparameters *exclude* for the robot train 3 and test 6 datasets.**Table 3**
Overview of different fusion combinations.

Absolute Pose	Relative Pose	Fusion
SfM	RPR	PGO
SfM	RPR (pre-trained)	PGO
APR	RPR	PGO
APR	RPR (pre-trained)	PGO
APR (pre-trained)	RPR	PGO
APR (pre-trained)	RPR (pre-trained)	PGO
SfM	RPR	Recurrent model
SfM	RPR (pre-trained)	Recurrent model
APR	RPR	Recurrent model
APR	RPR (pre-trained)	Recurrent model
APR (pre-trained)	RPR	Recurrent model
APR (pre-trained)	RPR (pre-trained)	Recurrent model

as shown in Figure 5g and Figure 5r. Finally, we captured the motion dynamics of two individuals randomly walking in the environment, with four absorber walls present (see trajectories in Figure 5d, 5e, 5o, and 5p).

The motion dynamics between the robot and humans are different. Cross-validation is conducted for all training and test datasets. Figure 6 presents images that are challenging with respect to the localization task, including the featureless structure of the absorber walls (Figure 6a and Figure 6b), which makes it particularly hard for SfM to extract features and match image points, or different scalings between distant warehouse racks (Figure 6d) and the small-scale labyrinth (Figure 6e). See Zangeneh et al. (2023), for challenges under ambiguous scenes.

(a) Median position error in m . (b) Median orientation error in $^{\circ}$.**Figure 9:** Evaluation of all SfM point clouds (for different hyperparameters, see x-axis) for the handheld train 4 and test 7 datasets. The dashed lines indicate the lowest error.(a) Median position error in m . (b) Median orientation error in $^{\circ}$.**Figure 10:** Evaluation of the best SfM point cloud (from Figure 9) for different hyperparameters *exclude* for the handheld train 4 and test 7 datasets.

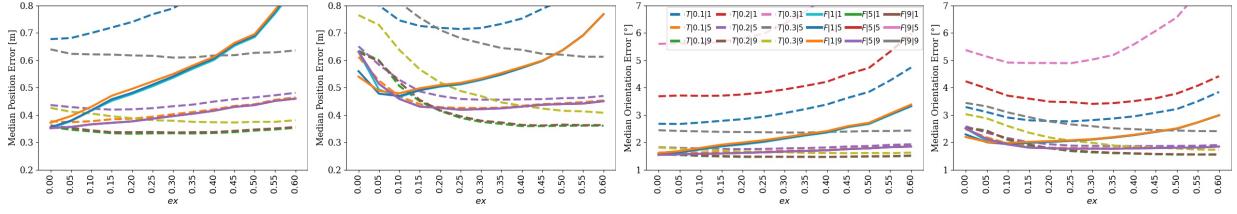
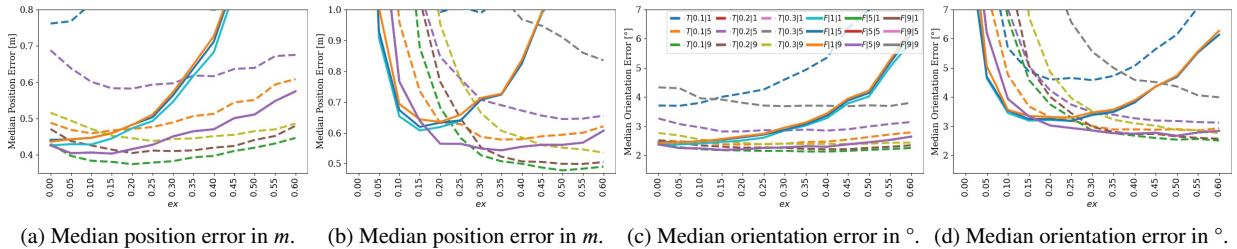
4.2. Overview of Experiments

Table 3 provides a comprehensive summary of the experiments conducted in our study. Our approach to estimating absolute pose involves either applying SfM (see Section 3.1) or using the time-distributed APR model (see Section 3.2). To predict relative poses, we utilize the RPR model (see Section 3.3). We evaluate the performance of both APR and RPR models with and without pre-training on the synthetically generated dataset (see Section 3.6). In terms of fusion, we compare our proposed fusion framework with eight different recurrent cells, namely LSTM, GRU, MGU, RAN, SRU, QRNN, TRNN, and CFN (see Section 3.5), against the state-of-the-art PGO technique (see Section 3.4).

4.3. Hardware Setup & Evaluation Metrics

For all experiments, we use Nvidia Tesla V100-SXM2 GPUs with 32 GB VRAM equipped with Core Xeon CPUs and 192 GB RAM. We use the Adam optimizer with a learning rate of 10^{-4} . We run each experiment for epochs = (iterations \cdot BS)/dataset_size, where we set iterations to 150,000 for APR and RPR and to 75,000 for the fusion models, the batch size BS is 50 for APR and RPR and 100 for the fusion models, and the dataset size depends on the scenario according to Table 2. We report results for the last epoch. For the evaluation of absolute predictions, we report the median absolute position in m and the median absolute orientation in $^{\circ}$.

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

(a) Median position error in m . (b) Median position error in m . (c) Median orientation error in $^{\circ}$. (d) Median orientation error in $^{\circ}$.**Figure 11:** SfM hyperparameter search for the robot train 3 and test 6 datasets. Figures a) and c) are with the three-point criterion, and Figures b) and d) are without the three-point criterion. The legend is equal for all subplots.(a) Median position error in m . (b) Median position error in m . (c) Median orientation error in $^{\circ}$. (d) Median orientation error in $^{\circ}$.**Figure 12:** SfM hyperparameter search for the handheld train 4 and test 7 datasets. Figures a) and c) are with the three-point criterion, and Figures b) and d) are without the three-point criterion. The legend is equal for all subplots.

5. Evaluation

5.1. Hyperparameter Search for SfM

In the first step, we conduct an extensive hyperparameter search for the SfM parameters described in Table 1 using two datasets: a dataset (train3 and test 6) and a handheld dataset (train 4 and test 7). We select the best hyperparameter combinations for point cloud reconstruction and apply them to the remaining robotic and handheld datasets. Each dataset (robotic and handheld) comprises a total of 1,752 parameters. Figure 7 illustrates all hyperparameter results for the robot dataset, while we select the best point clouds and evaluate the parameter *exclude* with values in $\{0.0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6\}$ in Figure 8. We observe that the parameter *exclude* has a significant impact on the position error, with values ranging from $0.24m$ to over $0.7m$. Based on Figure 8, we find that the parameter $ex = 0.30$ yields a low position error of $0.242m$, while the parameter $ex = 0.35$ yields a low orientation error of 1.19° . The hyperparameter search results for the handheld dataset are shown in Figure 9 and Figure 10. The error in this case increases due to the higher dynamics of the human, resulting in a position error of $0.37m$ and orientation error of 2.14° . The optimal value for the parameter *exclude* varies with the position (with $ex = 0.20$ being optimal) and orientation (with $ex = 0.35$ being optimal). To obtain highly accurate orientation predictions, a point cloud with a few and very accurate points is preferred, while a dense point cloud is better for low position error.

Figure 11 and Figure 12 present the selective search for the parameters of the three-point criterion for the robot dataset and the handheld dataset, respectively. We observe

Table 4

Number of matches per point for the eight training datasets (average and standard deviation).

Dataset	# Matches	Dataset	# Matches
Train 1	9.23 ± 8.41	Train 5	6.86 ± 4.50
Train 2	8.57 ± 6.93	Train 6	7.83 ± 5.73
Train 3	7.72 ± 6.51	Train 7	7.12 ± 4.72
Train 4	7.84 ± 6.29	Train 8	7.89 ± 6.42

that the three-point criterion results in decreased performance (Figures a and c represent the results with the criterion, while Figures b and d show the results without it). Additionally, we explore three other parameters (refer to the legend in Figure c). The first parameter is a hard limit to exclude points, which can be set to either *True* or *False*. We observe an improvement in results when using the limit. If the limit is used, the second parameter specifies the percentage of lower values to remove from $[0.1, 0.2, 0.3]$, with a preference for lower values. If the limit is set to *False*, we search for the minimum number of neighbors required to not exclude a point, with values in the range $[1, 5, 9]$. The best results are obtained when the number of neighbors is set to 5. The third parameter is the radius, which determines when a point counts as a neighbor for another point, with values in the range $[1, 5, 9]$. Here, a high value is preferred.

The evaluation of additional hyperparameters is presented in Appendix A.1. Figure 27 and Figure 28 show the results for the robot dataset, while Figure 29 and Figure 30 present the results for the handheld dataset. We select the best parameters for fusing with the RPR model.

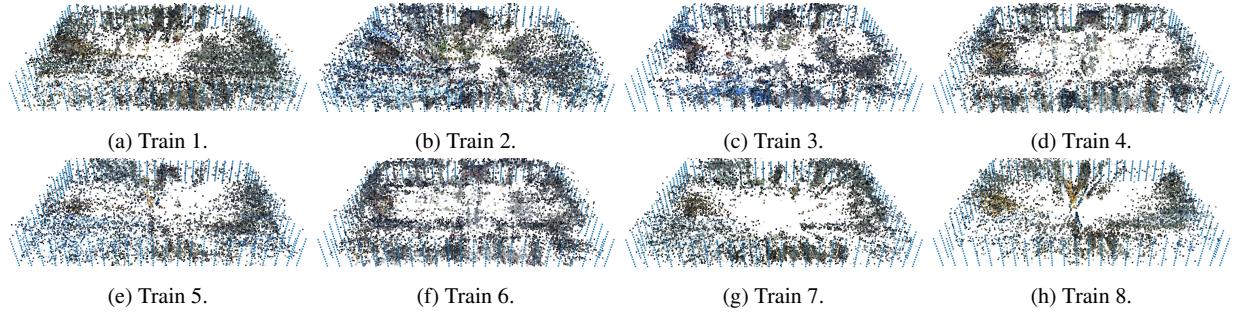


Figure 13: Overview of SfM point clouds for all eight training datasets. Blue dotted lines indicate the borders of the environment.

Table 5

Evaluation results for SfM. Results are shown as median position error in m and median orientation error in $^\circ$. We select the hyperparameter $ex = 0.3$ for the robotic datasets and the hyperparameter $ex = 0.2$ for the handheld datasets. The cell color indicates the relatedness between training and test environments and dynamics (light gray $\hat{=}$ similar environments, dark gray $\hat{=}$ strong environmental differences).

	Train 1		Train 2		Train 3		Train 4		Train 5		Train 6		Train 7		Train 8	
Test 1	0.4406	1.47	0.6720	2.18	0.6820	2.40	1.1758	31.83	1.7932	6.55	0.6600	2.30	2.0009	6.06	6.6378	20.30
Test 2	0.5736	1.94	0.5984	2.14	0.4296	1.71	1.1756	4.02	1.0385	3.64	0.7775	2.69	2.6908	8.47	5.9536	19.22
Test 3	0.8155	2.90	0.7514	2.78	0.4529	1.89	1.7175	6.40	1.5590	5.62	1.1602	3.91	3.4231	11.23	6.5891	21.41
Test 4	1.6407	5.50	0.6393	2.48	0.3407	1.54	1.8450	6.79	1.4746	5.40	1.1674	4.03	7.2445	26.19	7.0458	22.88
Test 5	6.5806	20.27	0.1312	0.82	0.2076	1.05	1.4789	4.79	0.5336	1.85	4.0145	12.14	5.7685	19.70	9.4618	29.55
Test 6	3.8720	13.03	0.7598	2.93	0.2417	1.19	0.5971	2.25	0.5417	1.99	4.3004	13.06	6.7546	23.65	10.3992	35.50
Test 7	24.5902	94.50	15.2681	64.65	1.5336	6.05	0.3742	2.17	0.3458	1.99	21.3927	80.61	18.3089	75.64	20.9043	79.34
Test 8	21.6899	81.23	12.6190	43.84	1.0662	4.41	0.4205	2.22	0.3049	1.82	10.3005	35.54	11.2760	42.33	14.2804	52.24
Test 9	1.0352	3.99	2.0459	6.84	0.8040	3.81	5.9806	18.24	1.0407	4.30	0.5175	2.52	1.5983	5.55	3.8926	12.44
Test 10	1.1925	5.56	5.6020	17.17	1.1586	5.47	2.8885	10.17	1.6000	6.32	0.5515	3.40	0.7954	4.14	4.9886	16.21
Test 11	12.1940	39.80	4.1512	14.22	0.4923	3.27	1.9017	7.29	0.9849	4.58	0.9164	4.34	1.8022	6.10	2.1391	7.25

5.2. Evaluation Results: SfM

We visualize the resulting SfM point clouds for all training datasets in Figure 13 with the dashed blue lines as the borders of the environment. The open environment of the train 1 dataset with no objects allows for a dense reconstruction, while train 2 also yields a high-density point cloud with wall features. However, on the top right and bottom left sections of the hall, points are missing due to the underrepresentation of images in these areas. However, datasets train 3, train 7, and train 8 contain many object features, while train 6 has more ceiling points. Dataset train 4, recorded with slow motions from person 1, has many feature-rich object points, while train 5, with fast motions from person 2, has a higher noise of points due to motion blur. The average number of matches per point cloud, reported in Table 4, supports these findings. Train 1 has the most matches with an average of 9.23 per point, resulting in a dense point cloud, while train 5 has a lower density with an average of 6.86 matches per point. The remaining datasets have an average number of matches ranging from 7.12 to 8.57.

Table 5 provides a summary of the SfM results, with the grey cells indicating the difference between the training and testing datasets. The evaluation shows that SfM is highly effective when tested on the same scenario as the training

scenario, as evidenced by the low errors on the train 2 and test 5 dataset (0.1312m and 0.82°) and the train 3 and test 6 dataset (0.2417m and 1.19°). For instance, with the point cloud from the train 3 dataset, the error increases consistent with increasing changes in the environment. Therefore, the error correlates with the changes in the environment (i.e., the color of the table cells). On the other hand, the prediction of orientation from the reconstruction is relatively robust against changes, with errors ranging from 1.19° on the test 6 dataset to 2.40° on the test 1 dataset. SfM also performs well on the handheld datasets (train 4 and train 5) with high motion dynamics, when evaluated on the same scenarios (test 7 and test 8). However, SfM fails when the reconstructions are applied to the robot evaluation datasets (and vice versa). In conclusion, while SfM is quite robust against environmental changes, it is not robust against dynamics (i.e., motion blur).

In Figure 14, we present three representative trajectory predictions from SfM. The trajectory predictions for all datasets can be found in Appendix A.2, spanning from Figure 31 to Figure 38. Overall, SfM produces few outliers (less than 100), in scenarios where the test images lack distinctive features. Moreover, SfM can accurately localize the robot in both open environments (see Figure 14a) and those with absorber walls (see Figure 14a). However, the

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

342

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

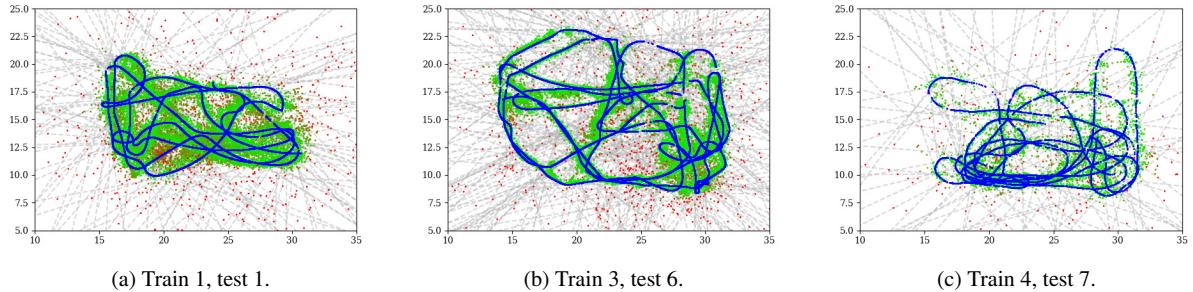


Figure 14: Evaluation of the predicted positions (green $\hat{=}$ low position error, red $\hat{=}$ large position error) against the ground truth trajectories (blue) for SfM.

Table 6

Evaluation results for the APR model. Results are shown as median position error in m and median orientation error in $^{\circ}$. We select the hyperparameter $ex = 0.3$ for the robotic datasets and the hyperparameter $ex = 0.2$ for the handheld datasets. The cell color indicates the relatedness between training and test environments and dynamics (light gray $\hat{=}$ similar environments, dark gray $\hat{=}$ strong environmental differences).

	Train 1	Train 2	Train 3	Train 4	Train 5	Train 6	Train 7	Train 8
Test 1	0.8076	2.12	3.0364	9.56	2.0245	10.92	4.4608	33.05
Test 2	1.0096	2.98	3.4245	10.36	2.0611	9.47	5.6366	34.33
Test 3	1.3636	4.72	3.0526	12.44	1.9051	9.32	4.5158	35.70
Test 4	2.2635	5.85	3.5413	9.95	1.9869	8.66	5.8072	29.51
Test 5	2.1091	6.93	0.7340	1.99	0.6417	2.32	3.9771	24.13
Test 6	3.2216	23.71	3.7292	19.39	0.4232	1.74	3.9909	16.07
Test 7	8.4233	79.54	7.1442	87.50	6.5239	39.58	0.6277	2.92
Test 8	6.2944	83.48	5.9621	82.11	4.6108	18.77	0.8599	3.88
Test 9	1.7769	5.63	3.9475	18.32	3.0726	21.53	4.6508	34.17
Test 10	4.1898	12.43	2.8045	37.11	4.2577	48.94	3.9608	41.43
Test 11	3.3845	16.54	4.3289	26.20	2.1233	8.81	3.9488	30.08

predicted trajectory may not be smooth in some cases (see Figure 14c).

5.3. Evaluation Results: APR and Augmented APR

In this section, an evaluation of the APR model is presented and the results are compared to SfM. The summary of the results is provided in Table 6. Similar to SfM, the APR model achieves the lowest position and orientation errors when evaluated in the same training and testing scenario. However, the errors produced by the APR model are higher compared to SfM (e.g., $0.4232m$ for APR and $0.2417m$ for SfM on the train 3 and test 6 dataset). The error increases significantly for changes in the environment (e.g., $2.0245m$ for the test 1 dataset), indicating that the APR model is less robust to changes compared to SfM. On the other hand, the APR model produces fewer outliers due to the restricted area in the learning process.

The motivation behind pre-training the APR model on a synthetically generated dataset is to enhance its generalization capabilities when objects are removed or added. We propose a comparison between the results of the APR model and the pre-trained APR model in Figure 15. The x-axis represents all possible combinations of training and testing

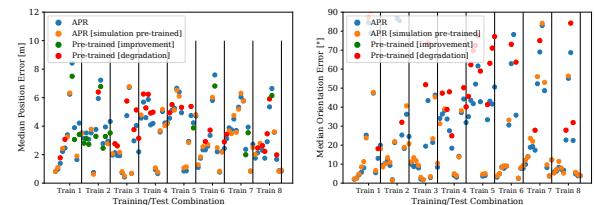


Figure 15: Comparison of APR with APR pre-trained on data generated from simulation.

scenarios. The green dots indicate an improvement in the error, while the red dots indicate a degradation in the error. The simulation environment contains many absorber walls that resemble the environment of dataset train 2, leading to a significant improvement in results for this scenario. However, the results decrease for the train 4 and train 5 datasets since the synthetic dataset lacks human dynamics and motion blur. When the predicted position error by APR is high, augmenting the data can further improve the results. However, the changes are marginal when the error is already low.

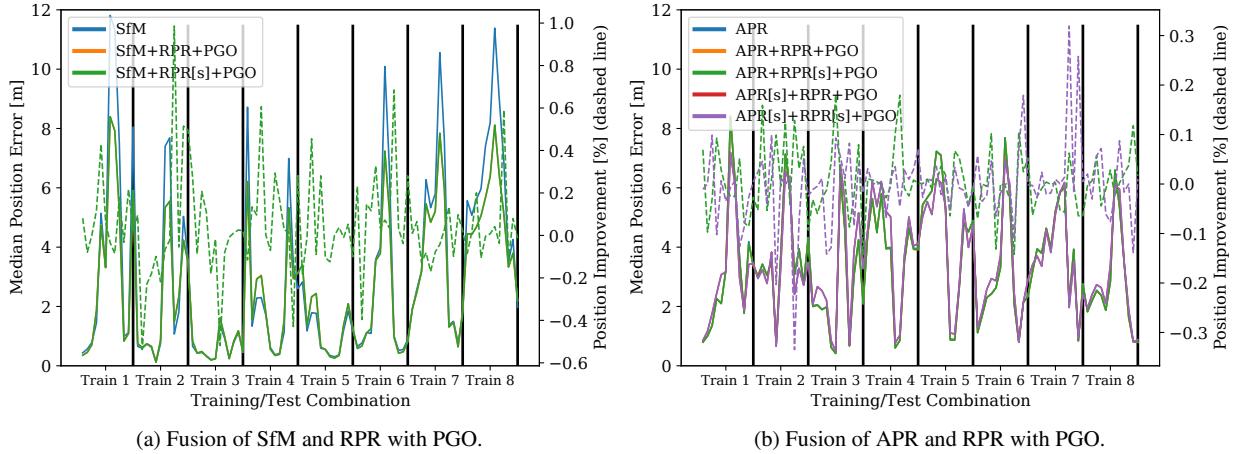


Figure 16: Comparison of SfM, respectively for APR, with the optimization of SfM, respectively of APR, with RPR and PGO. We evaluate APR and RPR pre-trained and non-pre-trained. Dashed lines show the position improvements in %.

5.4. Fusion Results

In the next step, we conduct an evaluation of the fusion of SfM and RPR, as well as the fusion of APR and RPR. The evaluation is carried out using PGO for pose refinement initially, followed by recurrent fusion cells. For an overview of the evaluation, please refer to Table 3.

PGO & Augmentation. Initially, we refine the absolute poses using relative poses with the state-of-the-art PGO algorithm. Figure 16a presents a comparison between the position error of SfM with the refined poses obtained from PGO (SfM+RPR+PGO). PGO has a significant positive impact on results in challenging scenarios but does not effect good localization results (e.g., train 2, train 3, and train 6 datasets). However, PGO marginally decreases the results for handheld datasets (train 4 and train 5). We also present the percentage improvement achieved by pre-training RPR with simulated data (dashed lines). While pre-training has a negative impact (-0.6%) on the train 2 and train 3 datasets, results can be slightly improved for the remaining datasets, up to 1.0%. Figure 16b presents results for refining APR with RPR. In contrast to SfM, this combination has no effect on results. Additionally, pre-training only leads to marginal improvements or decreases in results.

Recurrent APR-RPR Fusion & Augmentation. In order to fuse APR and RPR and predict an optimized absolute pose, it is essential to consider the specific deterministic, random-walk, non-linear, or long-memory behavior (or a combination of these) of both robotic and human motion. Specifically, the motion range of objects is limited to certain velocities and orientation changes. To address these challenges, a recurrent unit is necessary to process the required pose information. However, there is currently no clear understanding of which RNN-cell structure is most suitable for each type of behavior and the characterization of these units is not clear (Khaldi et al., 2023). Therefore, we provide a comprehensive evaluation of eight RNN units for the fusion

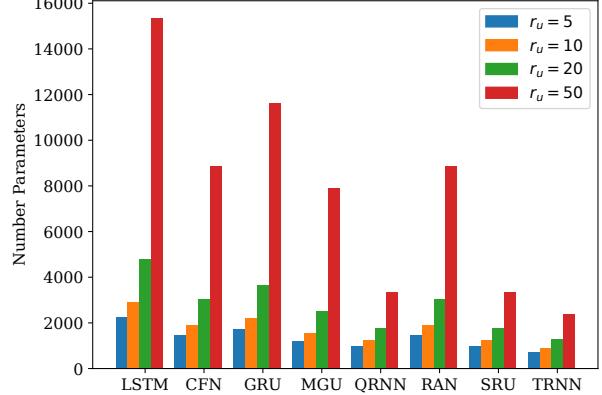


Figure 17: Comparison of trainable model parameters of all eight stacked recurrent networks for APR and RPR fusion for different number of units $r_u \in [5, 10, 20, 50]$.

task. According to Khaldi et al. (2023), an MGU cell is most suitable for a deterministic and non-linear behavior, while an LSTM cell is recommended for chaotic behavior, see also Yu et al. (2019). In order to further explore the optimal RNN-cell structure for the fusion task, we train and evaluate in addition to the commonly used cells LSTM, GRU, and MGU, the cell types CFN, QRNN, RAN, SRU, and TRNN. We vary the number of timestep input values ($N_t \in [3, 6, 10, 15, 25]$) and the number of recurrent units ($r_u \in [5, 10, 20, 50]$). Both single and two stacked RNN cells are evaluated using a copy memory task, which is designed to stress test the ability of recurrent networks to propagate long-term, distant information (Bai et al., 2018). This task assesses the model's capacity to retain information for different lengths of time (N_t). To compare the different models, Figure 17 shows the number of trainable model parameters for each of the eight stacked RNN cells and the number of units r_u . We observe that larger cell sizes (r_u) result in a significant increase in trainable parameters, leading to a

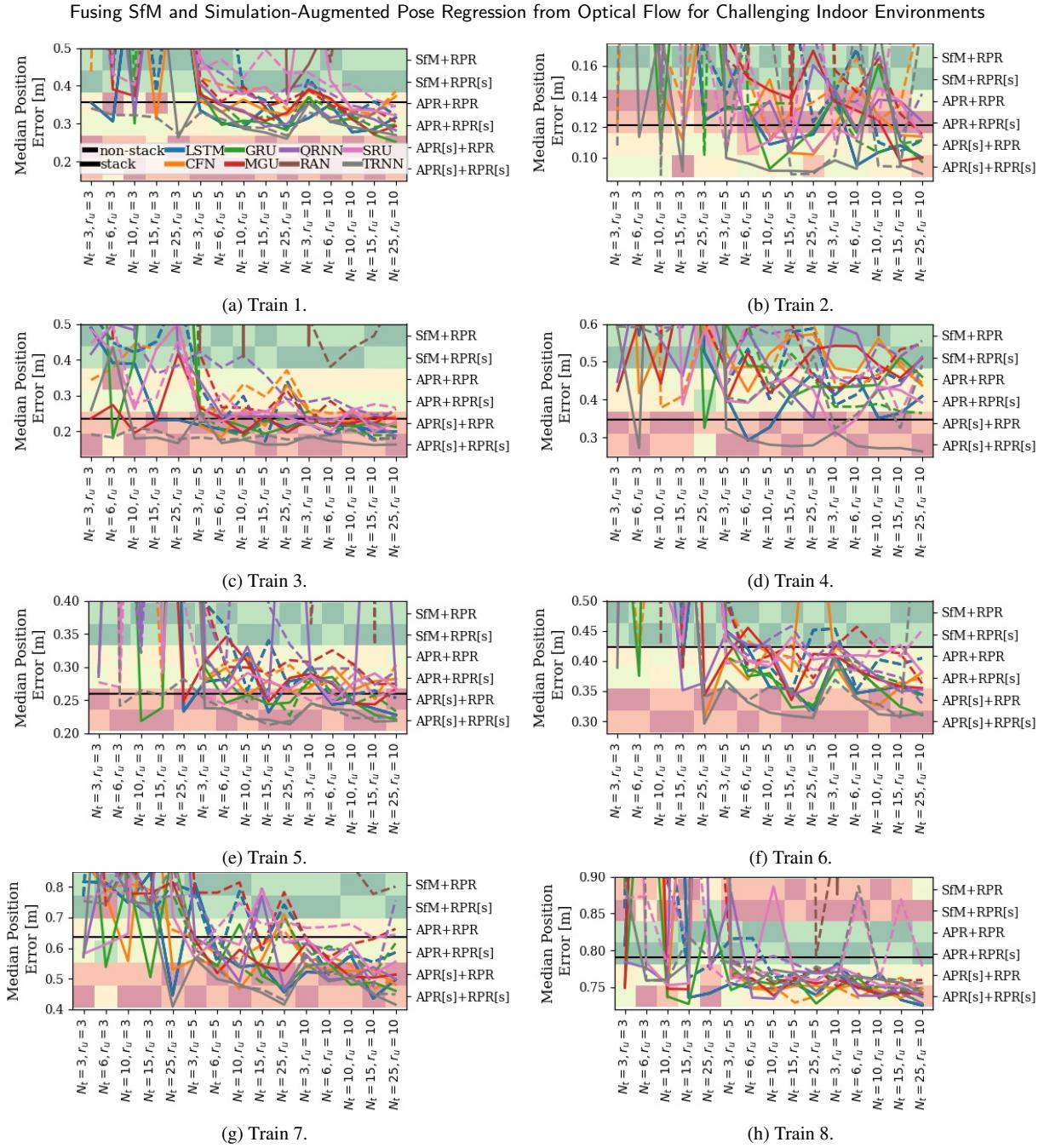


Figure 18: Evaluation of recurrent absolute (i.e., SfM or APR) and relative (i.e., RPR) pose fusion for eight training datasets evaluated on the corresponding testing dataset. Lines evaluate for recurrent cells and stacked (solid) or non-stacked (dashed) cells. The heatmap in the background ranks the best fusion method (green indicates best ranked method and red indicates the last ranked method). [s] indicates the simulation-augmented pre-training of APR or RPR. Black horizontal line indicates the optimization of SfM and RPR with PGO. For readability, we set y limits.

higher risk of overfitting. On the other hand, smaller gating mechanisms result in a decrease in the number of parameters (i.e., LSTM > GRU > MGU), with TRNN having the fewest parameters. Figure 18 provides an overview of the results for all eight training datasets and various fusions of absolute and relative models (with and without simulated pre-training),

compared to PGO (black line). We evaluate the parameters $N_t \in [3, 6, 10, 15, 25]$ and $r_u \in [3, 5, 10]$. We rank all methods with colored backgrounds. In the following, [s] indicates pre-training. The results of the method ranking indicate that SfM+RPR and SfM+RPR[s] outperform APR on all datasets, except for the train 8 dataset. The efficacy

Table 7

Evaluation results for the best recurrent cell for fusing SfM or APR with pre-trained RPR utilizing the stacked TRNN cell with N_t timesteps and r_u recurrent units. Results are shown as median position error in m and position improvement in %. The cell color indicates the relatedness between training and test environments and dynamics for the position error (light gray $\hat{=}$ similar environments, dark gray $\hat{=}$ strong environmental differences), and the degree of improvement (green $\hat{=}$ improvement, red $\hat{=}$ degradation) against SfM-only (see Table 5).

	Train 1	Train 2	Train 3	Train 4	Train 5	Train 6	Train 7	Train 8
Test 1	0.2606 +40.9	0.5909 +12.1	0.5249 +23.0	2.5297 -115.1	2.0405 -13.8	0.3877 +41.3	1.4115 +29.5	1.8464 +72.2
Test 2	0.3297 +42.5	0.4970 +16.9	0.3091 +28.0	1.7065 -45.1	1.8001 -73.3	0.4709 +39.4	1.7347 +35.5	2.1859 +63.3
Test 3	0.4724 +42.1	0.5831 +22.4	0.3335 +26.4	1.8748 -9.2	1.9356 -24.2	0.8070 +30.4	1.8795 +45.1	2.5305 +61.6
Test 4	0.9710 +40.8	0.4942 +22.7	0.2394 +29.7	1.8955 -2.7	1.9427 -31.7	0.7822 +33.0	3.5102 +51.5	2.3277 +66.9
Test 5	2.4365 +63.0	0.0888 +32.3	0.1399 +32.6	1.7744 -22.0	1.6923 -217.4	2.2221 +44.6	3.3965 +41.1	1.8474 +80.5
Test 6	1.9714 +49.1	0.6641 +12.6	0.1620 +33.0	1.5941 -184.1	1.6885 -211.7	2.5828 +39.9	3.6296 +46.3	2.7798 +73.3
Test 7	5.7052 +76.8	4.2081 +72.4	1.7238 -12.4	0.2631 +29.7	0.2612 +24.5	5.6267 +73.7	4.6363 +74.7	6.5534 +68.7
Test 8	4.2785 +80.3	4.8111 +61.9	1.8361 -61.7	0.3215 +23.5	0.2125 +30.3	3.5936 +65.1	4.0741 +63.9	5.3554 +62.5
Test 9	0.5803 +43.9	1.4832 +27.5	0.5536 +31.1	2.3406 +60.9	1.7902 -72.0	0.2952 +43.0	1.0633 +33.5	1.6314 +58.1
Test 10	0.6552 +45.1	2.9389 +47.5	0.7821 +32.5	1.9858 +31.3	1.8844 -17.8	0.3701 +32.9	0.4084 +43.7	0.7674 +84.6
Test 11	2.6793 +78.0	2.4830 +40.2	0.3220 +34.6	1.9044 -0.1	1.7822 -81.0	0.5500 +40.0	1.2626 +29.9	0.7451 +65.2

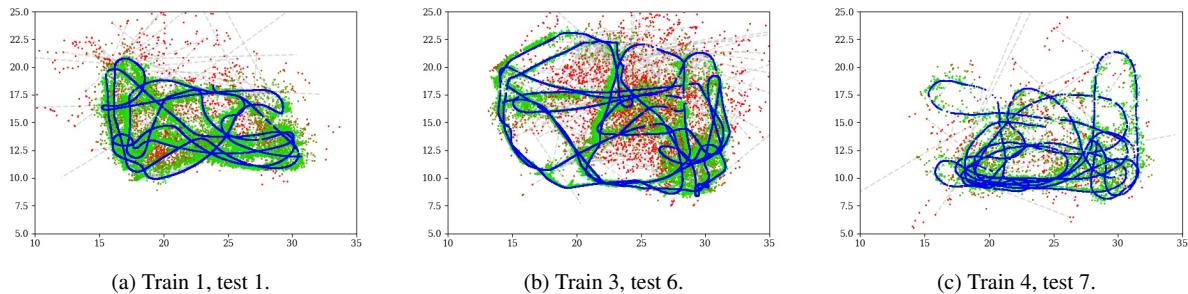


Figure 19: Evaluation of the predicted positions (green $\hat{=}$ low position error, red $\hat{=}$ large position error) against the ground truth trajectories (blue) for the recurrent fusion of SfM with the pre-trained RPR model utilizing the stacked TRNN cell with $N_t = 15$ timesteps and $r_u = 10$ recurrent units.

of pre-training is dependent on the dataset. Among the fusion models, most outperform PGO on the train 1, train 2, train 3, train 5, train 6, train 7, and train 8 datasets. Using two stacked RNN cells always yields better results than using only one cell. TRNN (Balduzzi and Ghifary, 2017) consistently achieves the lowest position errors, which is evident in Figures 18a, 18b, 18c, 18d, 18e, and 18f, and 18g. Therefore, a small model with few trainable parameters, i.e., $r_u = 10$ (as shown in Figure 17), is adequate for fusing absolute and relative poses. A large timestep size of $N_t = 25$ is observed to be better, as it enables the model to learn long-term dependencies of the motion dynamics. The difference between the recurrent units $r_u = 5$ and $r_u = 10$ is marginal and varies depending on the dataset, as evident in Figure 18d versus Figure 18h. A model size of $r_u = 3$ is too small to learn usable features.

Table 7 displays the improvement in position compared to SfM-only (Table 5) for all datasets. Based on the dataset used, the results can demonstrate an improvement ranging from +12.1% to +84.6% for the robotic dataset, especially when the training and testing are conducted on robotic datasets. However, the results show a substantial decline

for the handheld dataset when assessed on robotic datasets. This indicates that while recurrent cells can learn motion dynamics and produce a more precise predicted trajectory, they cannot adjust to new dynamics, such as transitioning from fast handheld movements to slow robotic movements. Nevertheless, the model can adapt from robotic to handheld movements. The enhancement is attributed to a significant decrease in outliers and the ability to predict smoothed trajectories, which is evident by comparing Figure 19 with Figure 14.

In Figure 20, we present additional results involving 17 convolutional, recurrent, and Transformer models, along with a comparison to PGO and TRNN (as shown in Figure 18). The hyperparameter searches are performed for the parameters N_t , r_u , nf , $depth$, d_{model} , and n_{layers} . Regarding the handheld dataset (Figure 20a), only TSiT (Zerveas et al., 2021) and gMLP (Liu et al., 2021) demonstrate superior performance compared to PGO and achieve similar results as TRNN. On the other hand, in the case of the robotic dataset (Figure 20b), most of the methods outperform PGO. However, TRNN exhibits lower positioning errors compared to all other methods, except for gMLP.

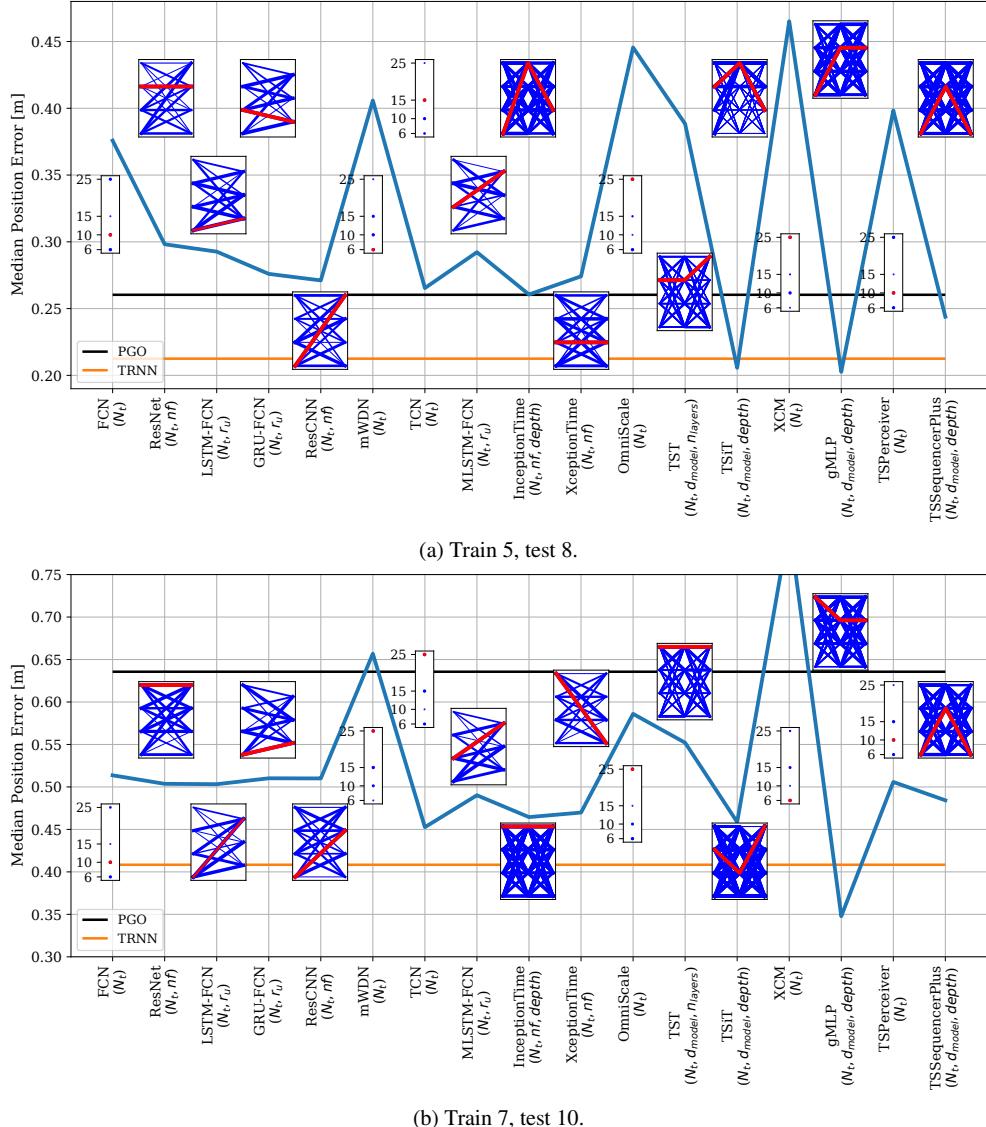


Figure 20: Hyperparameter search for the convolutional, recurrent convolutional, and Transformer models. We search for the hyperparameters $N_t \in [5, 10, 15, 25]$, $r_u \in [3, 5, 10]$, $n_f \in [16, 32, 64, 128]$, $depth \in [3, 4, 5, 6]$, $d_{model} \in [32, 64, 128, 256]$, and $n_{layers} \in [2, 3, 4, 5]$ (ordered from bottom to top). We select the best hyperparameters (marked red) to compare with PGO (black) and TRNN (orange).

5.5. Comparison to State-of-the-art

In comparison to state-of-the-art methods, we employ the accelerated coordinate encoding (ACE) technique proposed by Brachmann et al. (2023). ACE leverages a scene-agnostic feature backbone along with a scene-specific prediction head. In their work, Brachmann et al. (2023) utilize an MLP prediction head, enabling optimization across thousands of viewpoints simultaneously during each training iteration. This characteristics lead to a stable and rapid convergence. Notably, ACE represents the most recent advancement in this research domain and surpasses alternative approaches such as PoseNet (Kendall et al., 2015), MS-Transformer (Shavit et al., 2022), DSAC* (Brachmann and

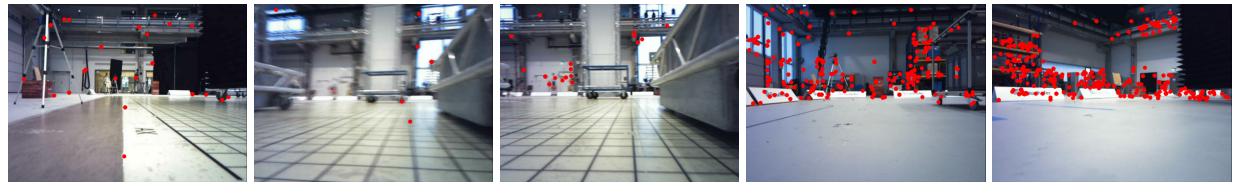
Rother, 2021), SANet (Yang et al., 2019), and SRC (Dong et al., 2022).

While ACE effectively produces a dense point cloud for nearby objects, such as the large black absorber walls, it fails to extract features from distant white walls and other feature-rich objects at a distance (see Figure 22). The pose prediction performance of ACE is notably diminished under challenging conditions, particularly when dealing with absorber walls. In contrast, our point clouds reconstructed with SfM exhibit more evenly distributed features, even from distant walls (see Figure 13). Furthermore, we provide a comparison of SfM+RPR fusion with TRNN networks and

Table 8

Comparison of results for the TRNN fusion model (left column) with ACE (Brachmann et al., 2023) (right column). Results are shown as median position error in m . The cell color indicates the relatedness between training and test environments and dynamics (light gray $\hat{=}$ similar environments, dark gray $\hat{=}$ strong environmental differences).

	Train 1	Train 2	Train 3	Train 4	Train 5	Train 6	Train 7	Train 8
Test 1	0.2606 0.271	0.5909 0.329	0.5249 0.492	2.5297 2.522	2.0405 1.701	0.3877 0.484	1.4115 1.619	1.8464 1.939
Test 2	0.3297 0.135	0.4970 0.510	0.3091 0.414	1.7065 1.349	1.8001 1.557	0.4709 0.559	1.7347 1.829	2.1859 2.130
Test 3	0.4724 0.166	0.5831 0.430	0.3335 0.369	1.8748 1.233	1.9356 1.369	0.8070 0.779	1.8795 1.714	2.5305 1.175
Test 4	0.9710 0.263	0.4942 0.495	0.2394 0.273	1.8955 1.044	1.9427 1.267	0.7822 0.755	3.5102 3.687	2.3277 0.958
Test 5	2.4365 2.715	0.0888 0.096	0.1399 0.114	1.7744 0.541	1.6923 0.618	2.2221 2.040	3.3965 3.583	1.8474 1.033
Test 6	1.9714 0.660	0.6641 0.687	0.1620 0.175	1.5941 0.830	1.6885 0.788	2.5828 2.235	3.6296 4.430	2.7798 1.612
Test 7	5.7052 3.550	4.2081 3.588	1.7238 0.418	0.2631 0.216	0.2612 0.227	5.6267 5.325	4.6363 5.317	6.5534 3.998
Test 8	4.2785 2.281	4.8111 1.130	1.8361 0.402	0.3215 0.325	0.2125 0.225	3.5936 3.645	4.0741 4.441	5.3554 2.686
Test 9	0.5803 0.319	1.4832 0.864	0.5536 0.977	2.3406 1.710	1.7902 1.894	0.2952 0.213	1.0633 1.014	1.6314 0.743
Test 10	0.6552 0.423	2.9389 1.198	0.7821 1.233	1.9858 1.561	1.8844 1.733	0.3701 0.535	0.4084 0.365	0.7674 0.431
Test 11	2.6793 2.676	2.4830 1.737	0.3220 0.426	1.9044 0.925	1.7822 1.526	0.5500 0.678	1.2626 1.239	0.7451 0.783



(a) Example 1. (b) Example 2. (c) Example 3. (d) Example 4. (e) Example 5.

Figure 21: Matches in red for five exemplary images.

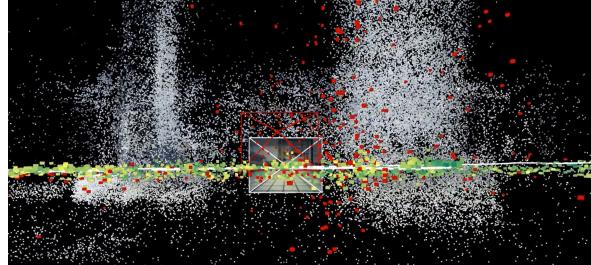


Figure 22: Retrieving absolute poses from the point cloud reconstructed with ACE (Brachmann et al., 2023).

ACE in Table 8. While ACE performs well for scenarios involving close objects, such as in the train 1 and test 3 dataset, our fusion model outperforms ACE in large-scale scenarios, as seen in the train 6 and test 1 dataset. Consequently, for further improvements in localization results, ACE can be used as a black-box model in combination with our relative module to achieve a more robust localization solution.

5.6. Robustness to Environmental Challenges

In this section, we aim to investigate the resilience of SfM against environmental variations and challenges. Figure 24 shows the position error for various environmental conditions. Figure 21 illustrates some sample images along with their corresponding matched pixels displayed as red points. One of the primary obstacles is coping with the noise in the input images caused by several factors such as lighting

conditions, motion blur, and camera distortion. As depicted in Figure 21b, the presence of motion blur in the image due to the rapid rotation of the robot results in only a few pixels being matched with other images. Conversely, in Figure 21c, a similar background context without motion blur allows for feature extraction from the surroundings, leading to successful matching. The correspondence between the position error in the x (top) and y (bottom) coordinates with respect to the orientation of the camera in the environment, as well as challenges present in the images, is visualized in Figure 23. The x and y-error exhibit similar behavior. The results indicate that the position error is low for the orientation of 180°, as the feature-rich warehouse racks are visible. However, the position error significantly increases for images that point to the right of the environment (0° and 360°). This is especially evident in the top figure for the timesteps 200 to 300 and 750 to 1,100, as many absorber walls and reflective surfaces are visible. Selected images recorded under optimal conditions (green) result in a low position error (e.g., at the timesteps 0 to 150 and 1,250 to 1,350). SfM is found to be robust against slight motion blur, as evident from timestep 2,600 to 2,750. The images in Figure 21d and Figure 21e contain rich information, leading to many pixels being matched. However, SIFT struggles to extract features from the black absorber walls present in Figure 21a and Figure 21e. Additionally, the images were generated using a fish-eye lens in the simulated environment, making it challenging to undistort them to construct an effective point cloud. Moreover, SfM techniques can encounter difficulties when dealing with

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

348

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

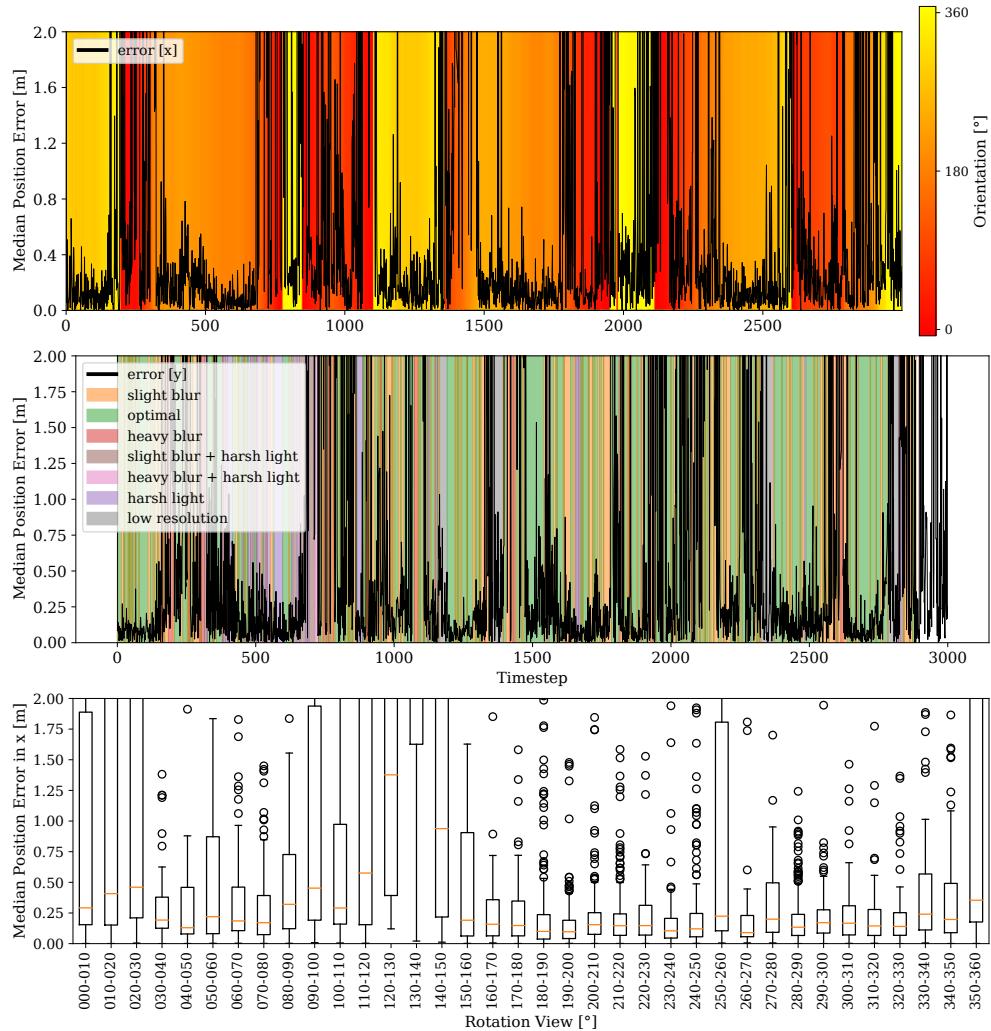


Figure 23: Plot of the positional error with respect to the x and y-coordinates of SfM for the train 4 dataset with environmental challenges marked in color. The top plot shows the orientation between 0° and 360°. The orientation of 0° points to the east of the environment. The middle plot shows seven challenges or optimal image conditions in color. The bottom plot shows the error dependent on the orientation in the environment.

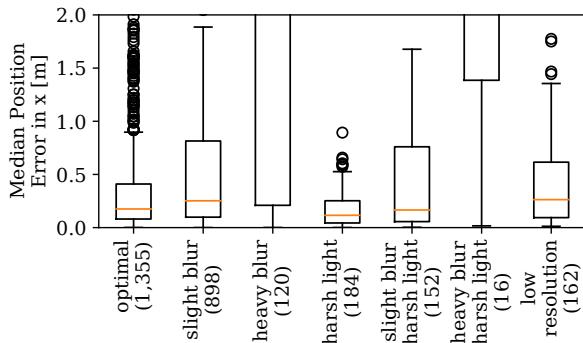


Figure 24: Position error of SfM for various environmental challenges. Number in brackets are the number of samples in the test set.

large-scale scenes. In such scenarios, the number of features and images can become too substantial for the algorithms to handle efficiently (Schönberger and Frahm, 2016). As a result, we were unable to construct point clouds from more than 1,000 images due to computation times on our hardware setup. Thus, we opted for 600 images for each dataset. One further challenge in SfM is dealing with moving objects in the scene. While our scenarios are static, there are moving objects present between the scenarios. Figure 25 illustrates the average number of matches per image view, which affects the point cloud density. A larger number of matches per view, up to 70 matches (as in Figure 25a), results in a denser point cloud (as in Figure 13a), whereas for the train 8 dataset, the point could be sparse (see Figure 13h) due to numerous objects and absorber walls and a smaller number of matches per view, up to 20, as in Figure 25h.

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

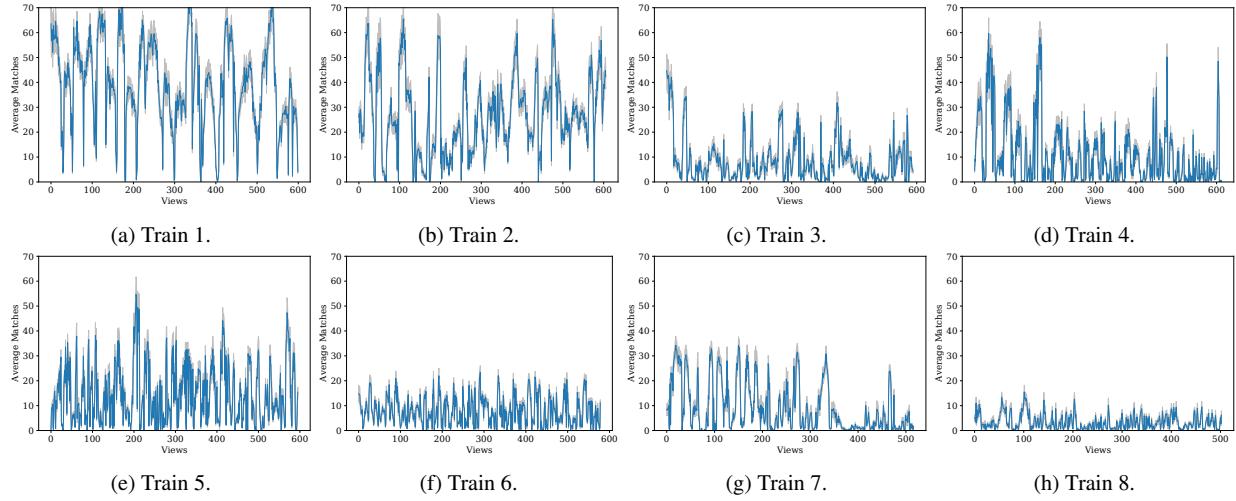


Figure 25: Average number of matches (y-axis) per view (x-axis) for all eight training datasets. Grey indicates standard deviation over all images.

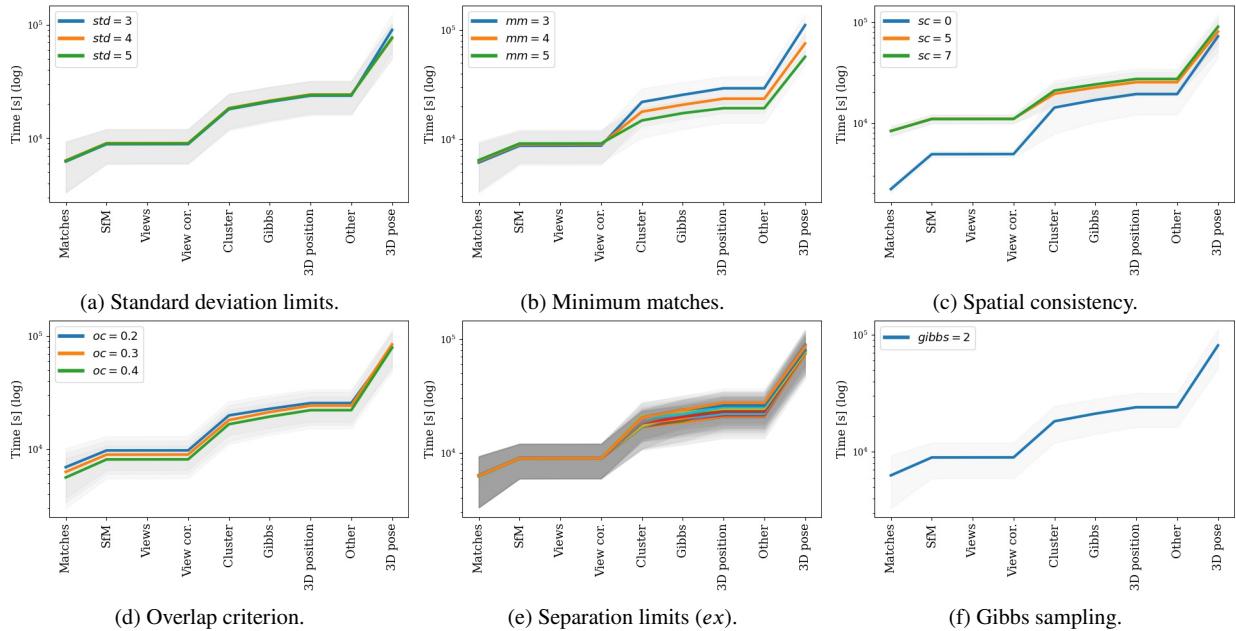


Figure 26: Overview of computation times (logarithmic in s) for different hyperparameter choices.

We observe that the number of matches varies significantly across different scenarios, indicating diverse environmental conditions in different areas.

In our analysis of the predicted trajectories (see the appendix, Figure 31 to Figure 38), we have observed that when constructing a point cloud from a clear environment (train 1) and evaluating it on environments with absorber walls, the reconstructed position exhibits an increase in error in those particular regions. This trend is noticeable in the top region of test 2, which contains one absorber wall (see Figure 31b), and the lower middle region of test 6, which

contains four absorber walls (see Figure 31e). This pattern is also evident in the other datasets we evaluated.

5.7. Computation Times

Figure 26 illustrates the computation times (in s) of SfM for point cloud reconstruction with respect to the parameters std , mm , sc , oc , ex , and $gibbs$. We highlight each parameter and average over the remaining parameters, displaying the standard deviation in grey. The time is displayed on a logarithmic scale. As seen in Figure 26a, the standard deviation (std) for point exclusion in BA has a negligible effect on computation time. The number of minimum matches ($mm \in$

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

350

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

{3, 4, 5}) has an impact on the time for cluster separation (as depicted in Figure 26b). Higher spatial consistency (sc) increases the time for match computation and view correction (as illustrated in Figure 26c) due to the use of k-means. The overlap criterion (oc) in Figure 26d and separation limits (ex) in Figure 26e have a minor impact on the training time. In general, cluster separation and 3D pose estimation from BA are the most time-consuming steps. SfM has the advantage of being able to run on a small CPU.

6. Conclusion

The focus of our study was on combining absolute poses obtained from SfM or an APR method with relative poses, in order to optimize and smoothen the trajectory of objects such as robots and humans. To estimate the relative pose, we proposed a recurrent network that learns the translation and rotation by analyzing the optical flow between consecutive images computed with the Lucas-Kanade algorithm. Our primary contribution was a fusion framework that uses eight different recurrent neural networks to combine absolute and relative poses, and we compared our approach to the state-of-the-art PGO technique for pose refinement. Additionally, we released a large visual database recorded in a challenging indoor environment that mimics warehouse scenarios, and we developed a simulation framework for generating synthetic images to pre-train APR and RPR models for faster training.

We conducted a comprehensive evaluation on the datasets, and the main findings can be summarized as follows: (1) The quality of the point cloud generated by SfM heavily depends on the hyperparameters selected, particularly the points that are removed from the point cloud. There is a cleavage of parameters on the position and orientation error. (2) A meticulous selection of these parameters results in significantly better outcomes compared to pose regression techniques. (3) SfM and APR methods experience difficulties in extracting features from challenging images, especially from feature-less images. However, APR methods exhibit more robustness against environmental changes. (4) Pre-training APR and RPR slightly enhances the position error by up to 1% for environments that resemble the simulated scenario. (5) PGO can refine the pose and improves high position errors, while simultaneously decreasing low position errors. (6) Our framework comprising various fusion cells based on convolutional, recurrent and Transformer models has demonstrated a significant enhancement in absolute pose error by facilitating smoother trajectory and higher resilience to challenges. This approach consistently outperforms PGO. The implementation of a strongly-typed RNN (TRNN) as a small-scale RNN model has yielded an average improvement of 42.67% in position results over the SfM-only results for the robotic datasets.

Acknowledgments

This work was supported by the Federal Ministry of Education and Research (BMBF) of Germany by Grant No.

01IS18036A (David Rügamer), as well as by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of “BAYERN DIGITAL II”.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data statement

For reproducibility and transparency, we publish our datasets and software on the following website: https://gitlab.cc-asp.fraunhofer.de/ottf/industry_datasets. The repository includes the eight training and 11 testing datasets, the selected images for structure from motion, the checkerboard images for calibration, the synthetically generated dataset, the point clouds recorded with the NavVis M4 system, and all ground truth poses. Additionally, we publish the source code. The repository is 798 GB in size in total.

CRediT authorship contribution statement

Felix Ott: conceptualization of this study, methodology, software, validation, formal analysis, investigation, data curation, writing – original draft, writing – review & editing, visualization, data recording, project administration. **Lucas Heublein:** conceptualization of this study, methodology, software, investigation, data curation. **David Rügamer:** formal analysis, writing – review & editing, visualization, supervision. **Bernd Bischl:** supervision. **Christopher Mutschler:** resources, writing – review & editing, supervision, funding acquisition.

References

Acharya, D., Tatli, C.J., Khoshelham, K., 2023. Synthetic-Real Image Domain Adaptation for Indoor Camera Pose Regression Using a 3D Model, in: , pp. 405–421. doi:[10.1016/j.isprsjprs.2023.06.013](https://doi.org/10.1016/j.isprsjprs.2023.06.013).

Bai, S., Kolter, J.Z., Koltun, V., 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, in: arXiv preprint arXiv:1803.01271v1 [cs.LG]. doi:[10.48550/arXiv.1803.01271](https://doi.org/10.48550/arXiv.1803.01271).

Baker, S., Matthews, I., 2004. Lucas-Kanade 20 Years on: A Unifying Framework, in: Intl. Journal of Computer Vision (IJCV), pp. 221–255. doi:[10.1023/B:VISI.0000011205.11775.f4](https://doi.org/10.1023/B:VISI.0000011205.11775.f4).

Baldazzi, D., Ghifary, M., 2017. Strongly-Typed Recurrent Neural Networks, in: Proc. of the Intl. Conf. on Machine Learning (ICML), New York, NY, pp. 1292–1300.

Bay, H., Tuytelaars, T., Gool, L.V., 2006. SURF: Speeded Up Robust Features, in: Europ. Conf. on Computer Vision (ECCV), pp. 404–417. doi:[10.1007/11744023_32](https://doi.org/10.1007/11744023_32).

Blanton, H., 2021. Revisiting Absolute Pose Regression, in: Dissertations, University of Kentucky.

Boitiaux, C., Marxer, R., Dune, C., Arnaubec, A., Hugel, V., 2022. Homography-Based Loss Function for Camera Pose Regression, in: IEEE Robotics and Automation Letters (RA-L), pp. 6242–6249. doi:[10.1109/LRA.2022.3168329](https://doi.org/10.1109/LRA.2022.3168329).

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

Brachmann, E., Cavallari, T., Prisacariu, V.A., 2023. Accelerated Coordinate Encoding: Learning to Relocalize in Minutes Using RGB and Poses, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR).

Brachmann, E., Rother, C., 2021. Visual Camera Relocalization from RGB and RGB-D Images Using DSAC, in: IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI), pp. 5847–5865. doi:10.1109/TPAMI.2021.3070754.

Bradbury, J., Merity, S., Xiong, C., Socher, R., 2017. Quasi-Recurrent Neural Networks, in: Intl. Conf. on Learning Representations (ICLR), Toulon, France.

Brahmbhatt, S., Gu, J., Kim, K., Hays, J., Kautz, J., 2018. Geometry-Aware Learning of Maps for Camera Localization, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, pp. 2616–2625. doi:10.1109/CVPR.2018.00277.

Brieger, T., Raichur, N.L., Jididi, D., Ott, F., Feigl, T., van der Merwe, J.R., Rügamer, A., Felber, W., 2022. Multimodal Learning for Reliable Interference Classification in GNSS Signals, in: Proc. of the Intl. Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+), Denver, CO, pp. 3210–3234. doi:10.33012/2022.18586.

Chidlovskii, B., Sadek, A., 2021. Adversarial Transfer of Pose Estimation Regression, in: Europ. Conf. on Computer Vision Workshops (EC-CVW). doi:10.1007/978-3-030-66415-2_43.

Chung, J., Gulcehre, C., Cho, K., Bengio, Y., 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, in: arXiv preprint arXiv:1412.3555.v1 [cs.NE]. doi:10.48550/arXiv.1412.3555.

Clark, R., Wang, S., Wen, H., Markham, A., Trigoni, N., 2017. VINet: Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem, in: Proc. of the AAAI Conf. on Artificial Intelligence (AAAI), pp. 3995–4001. doi:10.5555/3298023.3298149.

Costante, G., Ciarfuglia, T.A., 2018. LS-VO: Learning Dense Optical Subspace for Robust Visual Odometry Estimation, in: IEEE Robotics and Automation Letters (RA-L), pp. 1735–1742. doi:10.1109/LRA.2018.2803211.

Das, A., Dubbelman, G., 2018. An Experimental Study on Relative and Absolute Pose Graph Fusion for Vehicle Localization, in: IEEE Intelligent Vehicles Symposium (IV), Changshu, Suzhou, China, pp. 630–635.

Ding, X., Wang, Y., Tang, L., Jiao, Y., Xiong, R., 2020. Improving the Generalization of Network Based Relative Pose Regression: Dimension Reduction as a Regularizer, in: arXiv preprint arXiv:2010.12796.v1 [cs.CV]. doi:10.48550/arXiv.2010.12796.

Dong, S., Wang, S., Zhuang, Y., Kannala, J., Pollefeys, M., Chen, B., 2022. Visual Localization via Few-shot Scene Region Classification, in: IEEE Intl. Conf. on 3D Vision (3DV), Prague, Czech Republic. doi:10.1109/3DV57658.2022.00051.

Dosovitskiy, A., Fischer, P., Ilg, E., Häusser, P., Hazirbas, C., Golkov, V., van der Smagt, P., Cremers, D., Brox, T., 2015. FlowNet: Learning Optical Flow with Convolutional Networks, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV), Santiago de Chile, Chile, pp. 2758–2766. doi:10.1109/ICCV.2015.316.

Elsayed, N., Maida, A.S., Bayoumi, M., 2019. Deep Gated Recurrent and Convolutional Network Hybrid Model for Univariate Time Series Classification, in: Intl. Journal of Advanced Computer Science and Applications (IJACSA). doi:10.14569/IJACSA.2019.0100582.

Emter, T., Schirg, A., Wock, P., Petereit, J., 2019. Stochastic Cloning for Robust Fusion of Multiple Relative and Absolute Measurements, in: IEEE Intelligent Vehicles Symposium (IV), Paris, France.

Fauvel, K., Élisa Fromont, Masson, V., Faverdin, P., Termier, A., 2022. XEM: An Explainable-by-Design Ensemble Method for Multivariate Time Series Classification, in: WIREs Data Mining and Knowledge Discovery, pp. 917–957. doi:10.1007/s10618-022-00823-6.

Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F., 2020. InceptionTime: Finding AlexNet for Time Series Classification, in: WIREs Data Mining and Knowledge Discovery, pp. 1936–1962. doi:10.1007/s10618-020-00710-y.

Github (jahdiel), 2016. pySBA – Python Bundle Adjustment, in: Github: <https://github.com/jahdiel/pySBA>.

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J., 2009. A Novel Connectionist System for Unconstrained Handwriting Recognition, in: IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI), pp. 855–868. doi:10.1109/TPAMI.2008.137.

Han, L., Lin, Y., Du, G., Lian, S., 2019. DeepVIO: Self-supervised Deep Learning of Monocular Visual Inertial Odometry using 3D Geometric Constraints, in: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), Macau, China. doi:10.1109/IROS40897.2019.8968467.

Heck, J.C., Salem, F.M., 2017. Simplified Minimal Gated Unit Variations for Recurrent Neural Networks, in: IEEE Intl. Midwest Symposium on Circuits and Systems (MWSCAS), Boston, MA. doi:10.1109/MWSCAS.2017.8053242.

Hochreiter, S., Schmidhuber, J., 1997. Long Short-Term Memory, in: Neural Computation, pp. 1735–1780.

Idan, O., Shavit, Y., Keller, Y., 2023. Learning to Localize in Unseen Scenes with Relative Pose Regressors, in: arXiv preprint arXiv:2303.02717v1 [cs.CV]. doi:10.48550/arXiv.2303.02717.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T., 2017. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp. 1647–1655. doi:10.1109/CVPR.2017.179.

Iyer, G., Murthy, J.K., Gupta, G., Krishna, K.M., Paull, L., 2018. Geometric Consistency for Self-Supervised End-to-End Visual Odometry, in: Proc. of the IEEE/CVF Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT. doi:10.1109/CVPRW.2018.00064.

Jaegle, A., Borgeaud, S., Alayrac, J.B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., Hénaff, O., Botvinick, M.M., Zisserman, A., Vinyals, O., ao Carreira, J., 2021. Perceiver IO: A General Architecture for Structured Inputs & Outputs, in: Intl. Conf. on Learning Representations (ICLR).

Jiang, S., Jiang, C., Jiang, W., 2020. Efficient Structure from Motion for Large-scale UAV images: A Review and a Comparison of SfM Tools, in: ISPRS Journal of Photogrammetry and Remote Sensing (P&RS), pp. 230–251. doi:10.1016/j.isprsjprs.2020.04.016.

Karim, F., Majumdar, S., Darabi, H., Chen, S., 2017. LSTM Fully Convolutional Networks for Time Series Classification, in: IEEE Access, pp. 1662–1669. doi:10.1109/ACCESS.2017.2779939.

Karim, F., Majumdar, S., Darabi, H., Harford, S., 2019. Multivariate LSTM-FCNs for Time Series Classification, in: Neural Network, pp. 237–245. doi:10.1016/j.neunet.2019.04.014.

Kendall, A., Grimes, M., Cipolla, R., 2015. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV), Santiago de Chile, Chile, pp. 2938–2946. doi:10.1109/ICCV.2015.336.

Khaldi, R., Afia, A.E., Chiheb, R., Tabik, S., 2023. What is the Best RNN-cell Structure to Forecast Each Time Series Behavior, in: Expert Systems with Applications. doi:10.1016/j.eswa.2022.119140.

Kim, D., Kim, J., 2023. CT-Loc: Cross-domain Visual Localization with a Channel-wise Transformer, in: Neural Networks, pp. 369–383. doi:10.1016/j.neunet.2022.11.014.

Kreuzig, R., Ochs, M., Mester, R., 2019. DistanceNet: Estimating Traveled Distance From Monocular Images Using a Recurrent Convolutional Neural Network, in: Proc. of the IEEE/CVF Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA. doi:10.1109/CVPRW.2019.00165.

Laurent, T., von Brecht, J., 2017. A Recurrent Neural Network Without Chaos, in: Intl. Conf. on Learning Representations (ICLR), Toulon, France.

Lee, K., Levy, O., Zettlemoyer, L., 2017. Recurrent Additive Networks, in: arXiv preprint arXiv:1705.07393.v2 [cs.CL]. doi:10.48550/arXiv.1705.07393.

Lei, T., Zhang, Y., Artzi, Y., 2017. Training RNNs as Fast as CNNs, in: arXiv preprint arXiv:1709.02755v3 [cs.CL]. doi:10.48550/arXiv.1709.02755.

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

352 Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

Li, Q., Cao, R., Zhu, J., Fu, H., Zhou, B., Fang, X., Jia, S., Zhang, S., Liu, K., Li, Q., 2023a. Learn Then Match: A Fast Coarse-to-fine Depth Image-based Indoor Localization Framework for Dark Environments via Deep Learning and Keypoint-based Geometry Alignment, in: . pp. 169–177. doi:[10.1016/j.isprsjprs.2022.10.015](https://doi.org/10.1016/j.isprsjprs.2022.10.015).

Li, X., Ling, H., 2022. GTCaR: Graph Transformer for Camera Relocalization, in: *Europ. Conf. on Computer Vision (ECCV)*, pp. 229–246. doi:[10.1007/978-3-031-20080-9_14](https://doi.org/10.1007/978-3-031-20080-9_14).

Li, Y., Cao, R., Liu, K., Li, Z., Zhu, J., Bao, Z., Fang, X., Li, Q., Huang, X., Qiu, G., 2023b. Structure-Guided Camera Localization for Indoor Environments, in: . pp. 219–229. doi:[10.1016/j.isprsjprs.2023.05.034](https://doi.org/10.1016/j.isprsjprs.2023.05.034).

Lin, Y., Liu, Z., Huang, J., Wang, C., Du, G., Bai, J., Lian, S., Huang, B., 2019. Deep Global-Relative Networks for End-to-End 6-DoF Visual Localization and Odometry, in: *Proc. of the Pacific Rim Intl. Conf. Artificial Intelligence (PRICAI)*, Cuvu, Fiji. pp. 454–467. doi:[10.1007/978-3-030-29911-8_35](https://doi.org/10.1007/978-3-030-29911-8_35).

Liu, H., Dai, Z., So, D.R., Le, Q.V., 2021. Pay Attention to MLPs, in: *Advances in Neural Information Processing Systems (NIPS)*.

Löffler, C., Riechel, S., Fischer, J., Mutschler, C., 2018. Evaluation Criteria for Inside-Out Indoor Positioning Systems Based on Machine Learning, in: *IEEE Intl. Conf. on Indoor Positioning and Indoor Navigation (IPIN)*, Nantes, France. pp. 1–8. doi:[10.1109/IPIN.2018.8533862](https://doi.org/10.1109/IPIN.2018.8533862).

Lowe, D.G., 2004. Distinctive Image Features from Scale-Invariant Keypoints, in: *Intl. Journal of Computer Vision (IJCV)*.

Lu, Y., Lu, G., 2019. Deep Unsupervised Learning for Simultaneous Visual Odometry and Depth Estimation, in: *IEEE Intl. Conf. on Image Processing (ICIP)*, Taipei, Taiwan. doi:[10.1109/ICIP.2019.8803247](https://doi.org/10.1109/ICIP.2019.8803247).

Mansur, S., Habib, M., Pratama, G.N.P., Cahyadi, A.I., Ardijanto, I., 2017. Real Time Monocular Visual Odometry using Optical Flow: Study on Navigation of Quadrotora's UAV, in: *Intl. Conf. on Science and Technology - Computer (ICST)*, Yogyakarta, Indonesia. pp. 122–126. doi:[10.1109/ICST.2017.8011864](https://doi.org/10.1109/ICST.2017.8011864).

Mirovski, M., Grimes, M.K., Malinowski, M., Hermann, K.M., Anderson, K., Teplyashin, D., Simonyan, K., Kavukcuoglu, K., Zisserman, A., Hadsell, R., 2018. Learning to Navigate in Cities Without a Map, in: *Advances in Neural Information Processing Systems (NIPS)*, pp. 2424–2435.

Mitsuki, Y., Ryogo, Y., Kanji, T., 2021. S3G-ARM: Highly Compressive Visual Self-Localization from Sequential Semantic Scene Graph Using Absolute and Relative Measurements, in: *arXiv preprint arXiv:2109.04569*, v2 [cs.CV]. doi:[10.48550/arXiv.2109.04569](https://doi.org/10.48550/arXiv.2109.04569).

do Monte Lima, J.P.S., Uchiyama, H., ichiro Taniguchi, R., 2019. End-to-End Learning Framework for IMU-based 6-DOF Odometry, in: *MDPI Sensors*, p. 3777. doi:[10.3390/s19173777](https://doi.org/10.3390/s19173777).

Muller, P., Savakis, A., 2017. Flowdometry: An Optical Flow and Deep Learning Based Approach to Visual Odometry, in: *Proc. of the IEEE/CVF Winter Conf. for Applications on Computer Vision (WACV)*, Santa Rosa, CA. pp. 624–631. doi:[10.1109/WACV.2017.75](https://doi.org/10.1109/WACV.2017.75).

Muller, P.M., Savakis, A., Ptucha, R., Melton, R., 2016. Optical Flow and Deep Learning Based Approach to Visual Odometry, in: *Thesis*, Rochester Institute of Technology, Department of Computer Engineering, Rochester, NY.

OpenCV, 2022. Introduction to SIFT (Scale-Invariant Feature Transform), in: *OpenCV*: https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html.

Ott, F., Feigl, T., Löffler, C., Mutschler, C., 2020. ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization, in: *Proc. of the IEEE/CVF Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seattle, WA. pp. 187–198. doi:[10.1109/CVPRW50498.2020.00029](https://doi.org/10.1109/CVPRW50498.2020.00029).

Ott, F., Raichur, N.L., Rügamer, D., Feigl, T., Neumann, H., Bischl, B., Mutschler, C., 2022a. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression, in: *arXiv preprint arXiv:2208.00919*. doi:[10.48550/arXiv.2208.00919](https://doi.org/10.48550/arXiv.2208.00919).

Ott, F., Rügamer, D., Heublein, L., Bischl, B., Mutschler, C., 2022b. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach, in: *Proc. of the IEEE/CVF Winter Conf.* for Applications on Computer Vision (WACV), Waikoloa, HI. pp. 266–276. doi:[10.1109/WACV51458.2022.00131](https://doi.org/10.1109/WACV51458.2022.00131).

Pepe, A., Lasenby, J., 2023. CGA-PoseNet: Camera Pose Regression via a 1D-Up Approach to Conformal Geometric Algebra, in: *arXiv preprint arXiv:2302.05211v1 [cs.CV]*.

Piasco, N., Sidibé, D., Demonceaux, C., Gouet-Brunet, V., 2018. A Survey on Visual-based Localization: On the Benefit of Heterogeneous Data, in: *Pattern Recognition*, pp. 90–109. doi:[10.1016/j.patcog.2017.09.013](https://doi.org/10.1016/j.patcog.2017.09.013).

Qiao, C., Xiang, Z., Fan, Y., Bai, T., Zhao, X., Fu, J., 2023. TransAPR: Absolute Camera Pose Regression with Spatial and Temporal Attention, in: *IEEE Robotics and Automation Letters (RA-L)*, pp. 4633–4640. doi:[10.1109/LRA.2023.3286123](https://doi.org/10.1109/LRA.2023.3286123).

Radanovic, M., Khoshelham, K., Fraser, C., 2023. Aligning the Real and the Virtual World: Mixed Reality Localisation Using Learning-based 3D-3D Model Registration, in: . doi:[10.1016/j.aei.2023.101960](https://doi.org/10.1016/j.aei.2023.101960).

Radwan, N., Valada, A., Burgard, W., 2018. VLocNet++: Deep Multitask Learning for Semantic Visual Localization and Odometry, in: *IEEE Robotics and Automation Letters (RA-L)*, pp. 4407–4414. doi:[10.1109/LRA.2018.2869640](https://doi.org/10.1109/LRA.2018.2869640).

Rahimian, E., Zabihi, S., Atashzar, S.F., Asif, A., Mohammadi, A., 2019. XceptionTime: A Novel Deep Architecture based on Depthwise Separable Convolutions for Hand Gesture Classification, in: *arXiv preprint arXiv:1911.03803v1 [cs.LG]*.

Resch, B., Lensch, H.P.A., Wang, O., Pollefeys, M., Sorkine-Hornung, A., 2015. Scalable Structure from Motion for Densely Sampled Videos, in: *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA. pp. 3936–3944. doi:[10.1109/CVPR.2015.7299019](https://doi.org/10.1109/CVPR.2015.7299019).

Ruan, J., He, L., Guan, Y., Zhang, H., 2023. Combining Scene Coordinate Regression and Absolute Pose Regression for Visual Relocalization, in: *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, London, UK. doi:[10.1109/ICRA48891.2023.10160317](https://doi.org/10.1109/ICRA48891.2023.10160317).

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An Efficient Alternative to SIFT or SURF, in: *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, Barcelona, Spain. doi:[10.1109/ICCV.2011.6126544](https://doi.org/10.1109/ICCV.2011.6126544).

Sattler, T., Zhou, Q., Pollefeys, M., Leal-Taixé, L., 2019. Understanding the Limitations of CNN-based Absolute Camera Pose Regression, in: *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA. pp. 3302–3312. doi:[10.1109/CVPR.2019.00342](https://doi.org/10.1109/CVPR.2019.00342).

Schönberger, J.L., Frahm, J.M., 2016. Structure-from-Motion Revisited, in: *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV. doi:[10.1109/CVPR.2016.445](https://doi.org/10.1109/CVPR.2016.445).

Shavit, Y., Ferens, R., Keller, Y., 2022. Learning Multi-Scene Absolute Pose Regression with Transformers, in: *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, Montreal, QC. doi:[10.1109/ICCV48922.2021.00273](https://doi.org/10.1109/ICCV48922.2021.00273).

Shavit, Y., Keller, Y., 2022. Camera Pose Auto-encoders for Improving Pose Regression, in: *Europ. Conf. on Computer Vision (ECCV)*. doi:[10.1007/978-3-031-20080-9_9](https://doi.org/10.1007/978-3-031-20080-9_9).

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R., 2017. Learning from Simulated and Unsupervised Images Through Adversarial Training, in: *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2242–2251. doi:[10.1109/CVPR.2017.24](https://doi.org/10.1109/CVPR.2017.24).

Stahlke, M., Kram, S., Ott, F., Feigl, T., Mutschler, C., 2022. Estimating TOA Reliability with Variational Autoencoders, in: *IEEE Sensors Journal*, pp. 5133–5140. doi:[10.1109/JSEN.2021.3101933](https://doi.org/10.1109/JSEN.2021.3101933).

von Stumberg, L., Usenko, V., Cremers, D., 2018. Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization, in: *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Brisbane, Australia. doi:[10.1109/ICRA.2018.8462905](https://doi.org/10.1109/ICRA.2018.8462905).

von Stumberg, L., Wenzel, P., Yang, N., Cremers, D., 2020. LM-Reloc: Levenberg-Marquardt Based Direct Visual Relocalization, in: *IEEE Intl. Conf. on 3D Vision (3DV)*, Fukuoka, Japan. doi:[10.1109/3DV50981.2020.00107](https://doi.org/10.1109/3DV50981.2020.00107).

 Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going Deeper with Convolutions, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR), Boston, MA. pp. 1–9. doi:10.1109/CVPR.2015.7298594.

Tang, W., Long, G., Liu, L., Zhou, T., Blumenstein, M., Jiang, J., 2022. Omni-Scale CNNs: a Simple and Effective Kernel Size Configuration for Time Series Classification, in: Intl. Conf. on Learning Representations (ICLR).

Tatsunmai, Y., Taki, M., 2022. Sequencer: Deep LSTM for Image Classification, in: Advances in Neural Information Processing Systems (NIPS).

Valada, A., Radwan, N., Burgard, W., 2018. Deep Auxiliary Learning for Visual Localization and Odometry, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), Brisbane, Australia. pp. 6939–6946. doi:10.1109/ICRA.2018.8462979.

Venkataraman, H., 2020. 3D Reconstruction Using Structure from Motion, in: Github: <https://github.com/harish-vnkt/structure-from-motion>.

Wang, J., Qi, Y., 2023. Deep 6-DoF Camera Relocalization in Variable and Dynamic Scenes by Multitask Learning, in: Machine Vision and Applications. doi:10.1007/s00138-023-01388-0.

Wang, J., Wang, Z., Liu, J., Wu, J., 2018. Multilevel Wavelet Decomposition Network for Interpretable Time Series Analysis, in: ACM Intl. Conf. on Knowledge Discovery & Data Mining (SIGKDD). doi:10.1145/3219819.3220060.

Wang, S., Clark, R., Wen, H., Trigoni, N., 2017. DeepVO: Towards end-to-end Visual Odometry with deep Recurrent Convolutional Neural Networks, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), Singapore, Singapore. pp. 2043–2050. doi:10.1109/ICRA.2017.7989236.

Wang, Z., Yan, W., Oates, T., 2016. Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline, in: arXiv preprint arXiv:1611.06455v4 [cs.LG].

Winkelbauer, D., Denninger, M., Triebel, R., 2021. Learning to Localize in New Environments from Synthetic Training Data, in: arXiv preprint arXiv:2011.04539v2 [cs.RO]. doi:10.48550/arXiv.2011.04539.

Xu, M., Wang, Y., Xu, B., Zhang, J., Poslad, J.R.S., Xu, P., 2022. A Critical Analysis of Image-based Camera Pose Estimation Techniques, in: arXiv preprint arXiv:2201.05816v1 [cs.CV].

Yang, L., Bai, Z., Tang, C., Li, H., Furukawa, Y., Tan, P., 2019. SANet: Scene Agnostic Network for Camera Localization, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV), Seoul, Korea. doi:10.1109/ICCV.2019.00013.

Yang, N., von Stumberg, L., Wang, R., Cremers, D., 2020. D3VO: Deep Depth, Deep Pose and Deep Uncertainty for Monocular Visual Odometry, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR), Seattle, WA. doi:10.1109/CVPR42600.2020.00136.

Yu, H., Feng, Y., Ye, W., Jiang, M., Bao, H., Zhang, G., 2023. Improving Feature-based Visual Localization by Geometry-Aided Matching, in: arXiv preprint arXiv:2211.08712.v2 [cs.CV]. doi:10.48550/arXiv.2211.08712.

Yu, Y., Si, X., Hu, C., Zhang, J., 2019. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures, in: Neural Computation, pp. 1235–1270. doi:10.1162/neco_a_01199.

Zangeneh, F., Bruns, L., Dekel, A., Pieropan, A., Jensfelt, P., 2023. A Probabilistic Framework for Visual Localization in Ambiguous Scenes, in: arXiv preprint arXiv:2301.02086 [cs.CV]. doi:10.48550/arXiv.2301.02086.

Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C., 2021. A Transformer-based Framework for Multivariate Time Series Representation Learning, in: Proc. of the ACM Intl. Conf. on Knowledge Discovery & Data Mining (SIGKDD), pp. 2114–2124. doi:10.1145/3447548.3467401.

Zhan, H., Weerasekera, C.S., Bian, J.W., Reid, I., 2020. Visual Odometry Revisited: What Should Be Learned?, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), Paris, France. doi:10.1109/ICRA40945.2020.9197374.

Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O., 2017. Understanding Deep Learning (still) Requires Rethinking Generalization, in: Communications of the ACM, pp. 107–115. doi:10.1145/3446776.

Zheng, L., Yang, Y., Tian, Q., 2018. SIFT Meets CNN: A Decade Survey of Instance Retrieval, in: IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI), pp. 1224–1244.

Zhi-Yu, C., Po-Heng, C., Kuan-Wen, C., Chen-Yu, C., 2021. PA-FlowNet: Pose-Auxiliary Optical Flow Network for Spacecraft Relative Pose Estimation, in: Proc. of the IAPR Intl. Conf. on Pattern Recognition (ICPR). doi:10.1109/ICPR48806.2021.9413201.

Zhou, G.B., Wu, J., Zhang, C.L., Zhou, Z.H., 2016. Minimal Gated Unit for Recurrent Neural Networks, in: Intl. Journal of Automation and Computing (IJAC), pp. 226–234. doi:doi.org/10.1007/s11633-016-1006-2.

Zhou, L., Luo, Z., Shen, T., Zhang, J., Zhen, M., Yao, Y., Fang, T., Quan, L., 2020. KFNet: Learning Temporal Camera Relocalization using Kalman Filtering, in: Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR), Seattle, WA. pp. 4919–4928. doi:10.1109/CVPR42600.2020.00497.

Zou, X., Wang, Z., Li, Q., Sheng, W., 2019. Integration of Residual Network and Convolutional Neural Network Along with Various Activation Functions and Global Pooling for Time Series Classification, in: Neurocomputing, pp. 39–45. doi:10.1016/j.neucom.2019.08.023.

A. Appendix A

A.1. Hyperparameter Search

In Figure 27 and Figure 28, we conduct a detailed hyperparameter search for the robot dataset, while in Figure 29 and Figure 30, we perform a similar search for the handheld dataset. The x-axis of each figure represents a combination of three hyperparameters (ex_1, ex_2, ex_3) that control the separation limit. We search for each of these parameters in the range of [10, 20, 30, 40], subject to the constraints that $ex_1 \geq ex_2$, $ex_1 \geq ex_3$, and $ex_2 \geq ex_3$. Here, ex_1 is the cluster separation limit, ex_2 is the Gibbs separation parameter, and ex_3 is the 3D position adjustment of BA. We observe that certain combinations of these parameters result in high error rates, but higher values of the parameters generally yield lower position and orientation errors. For example, $(ex_1, ex_2, ex_3) = (40, 30, 30)$ produces the best results.

For each combination of hyperparameters, we generate a point cloud using the parameters $mm \in [3, 4, 5]$, $sl = std \in [3, 4, 5]$, the spatial consistency $sc \in [0, 1, 2]$, and the overlap criterion $oc \in [0.0, 0.1, 0.2]$. For the robot dataset, we found that the combination of $sc = 2$ and $oc = 2.0$ yields the best results (position error below 0.3m and orientation error below 1.5°), as depicted in Figure 27i and Figure 28i. For the handheld dataset, we recommend the combination of $sc = 1$ and $oc = 1.0$, refer to Figure 29e and Figure 30e. We also observe that a high value for the minimum matches parameter $mm = 5$ is clearly the best choice, while $std = [3, 4, 5]$ produces similar results.

A.2. Trajectory Plots

Figure 31 to Figure 38 contain visualizations of the predicted absolute poses for SfM, encompassing all eight training datasets and ten test datasets.

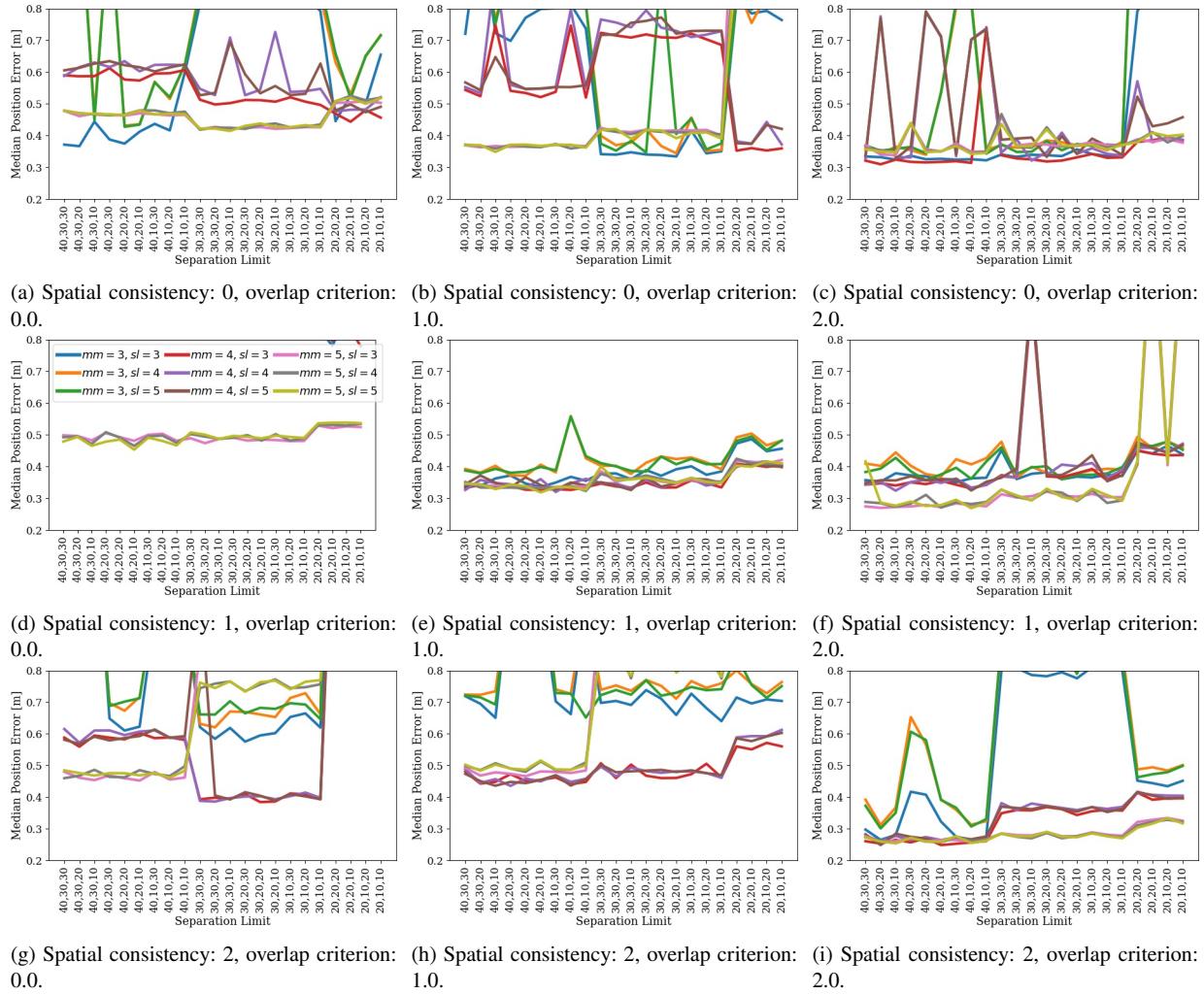


Figure 27: Detailed evaluation results for the SfM hyperparameter search for the robot train 3 and test 6 datasets. Median position error in m . For readability, the label spacing is fixed. The legend shows the *minimum matches* hyperparameter. The legend is equal for all subplots.

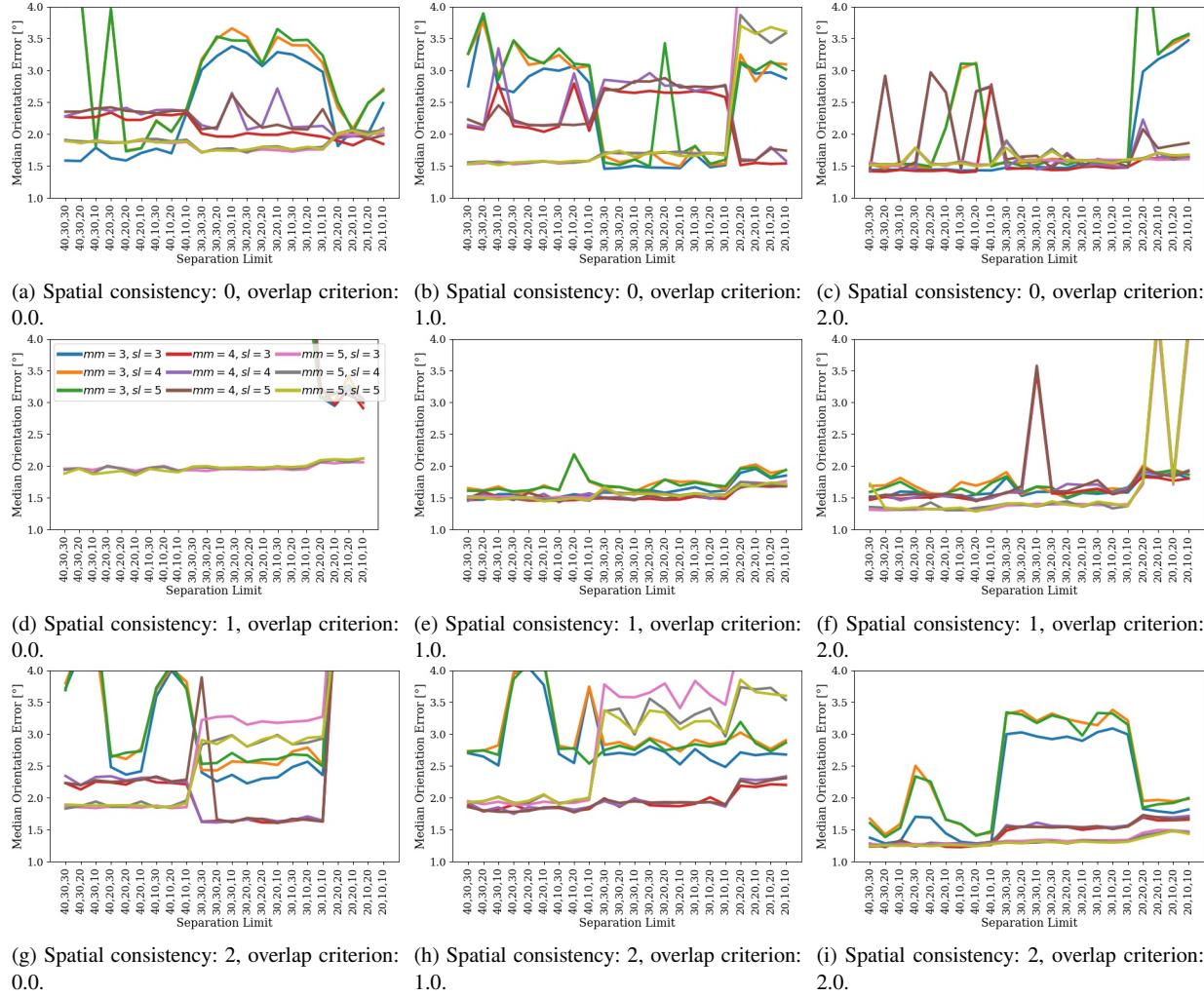


Figure 28: Detailed evaluation results for the SfM hyperparameter search for the robot train 3 and test 6 datasets. Median orientation error in °. For readability, the label spacing is fixed. The legend shows the *minimum matches* hyperparameter. The legend is equal for all subplots.

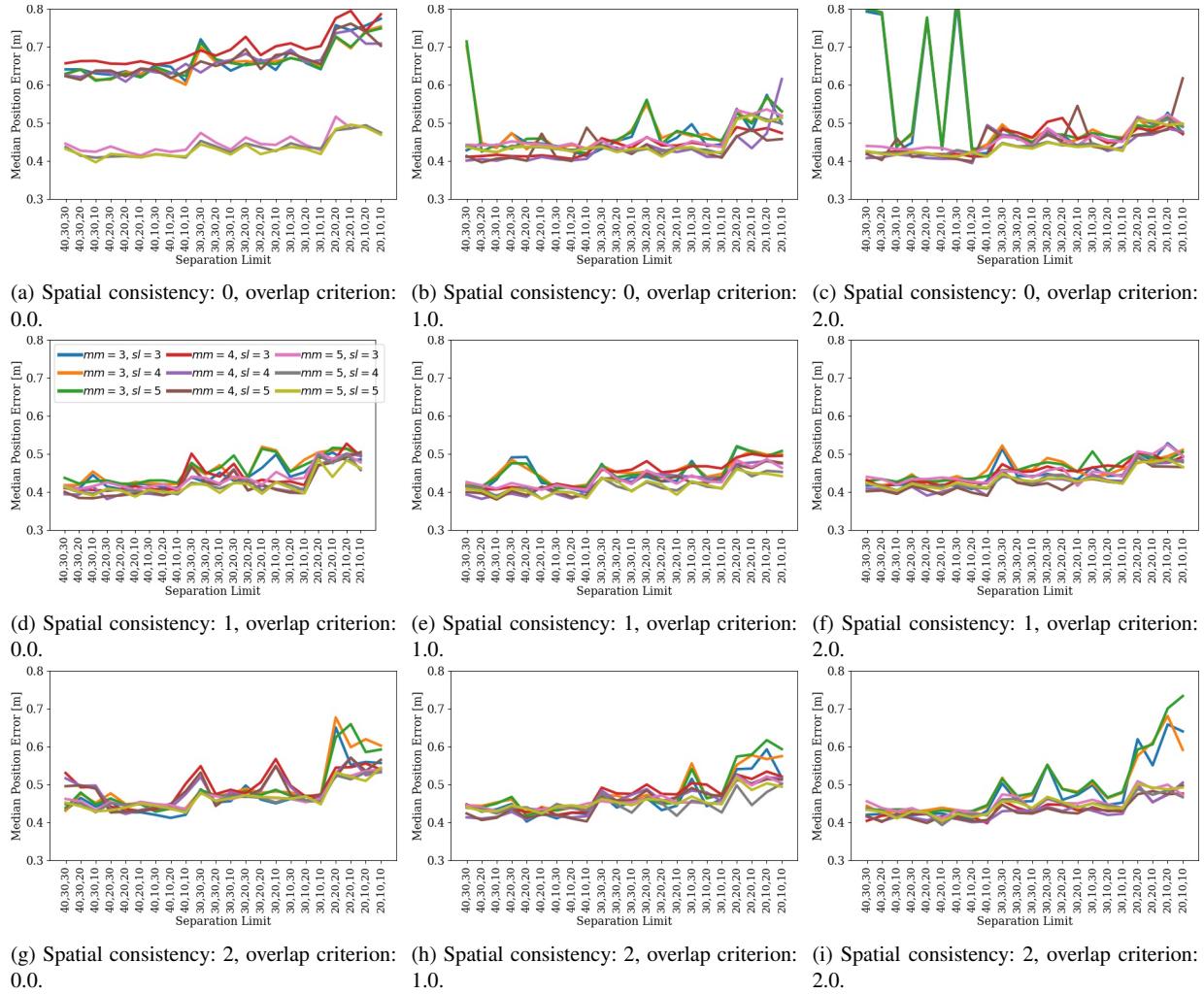


Figure 29: Detailed evaluation results for the SfM hyperparameter search for the handheld train 4 and test 7 datasets. Median position error in m . For readability, the label spacing is fixed. The legend shows the *minimum matches* hyperparameter. The legend is equal for all subplots.



Figure 30: Detailed evaluation results for the SfM hyperparameter search for the handheld train 4 and test 7 datasets. Median orientation error in °. For readability, the label spacing is fixed. The legend shows the *minimum matches* hyperparameter. The legend is equal for all subplots.

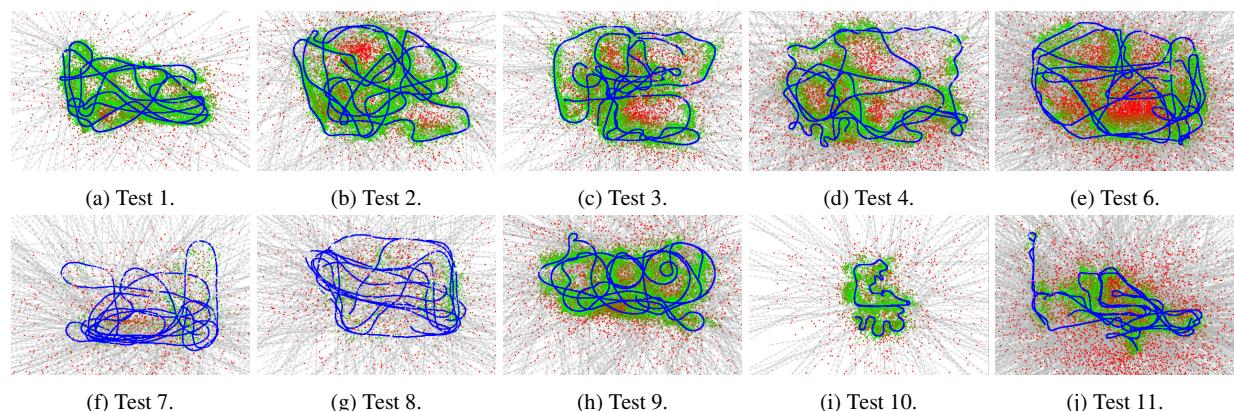


Figure 31: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 1 dataset.

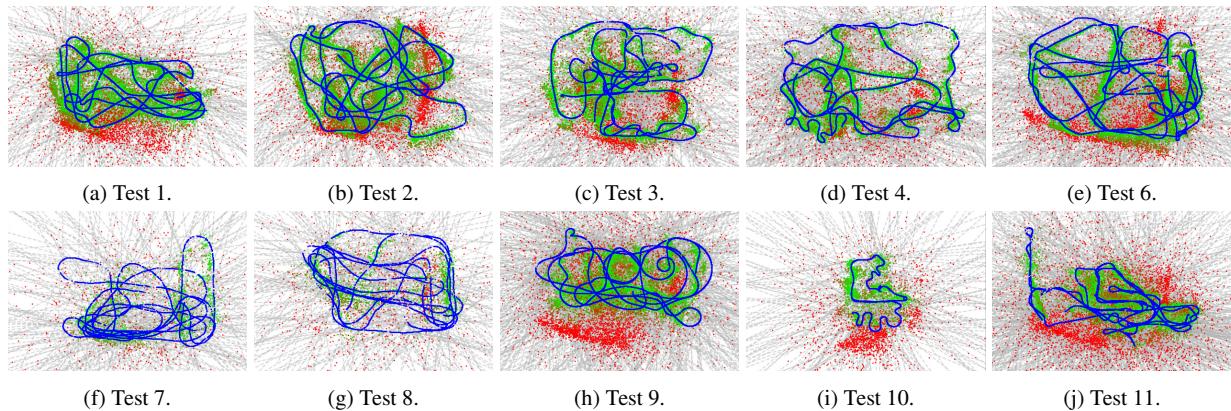


Figure 32: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 2 dataset.

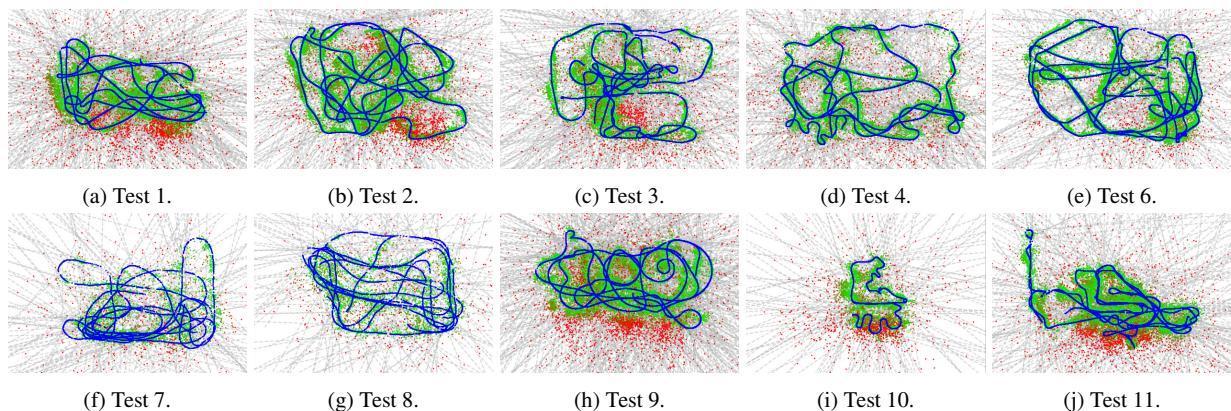


Figure 33: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 3 dataset.

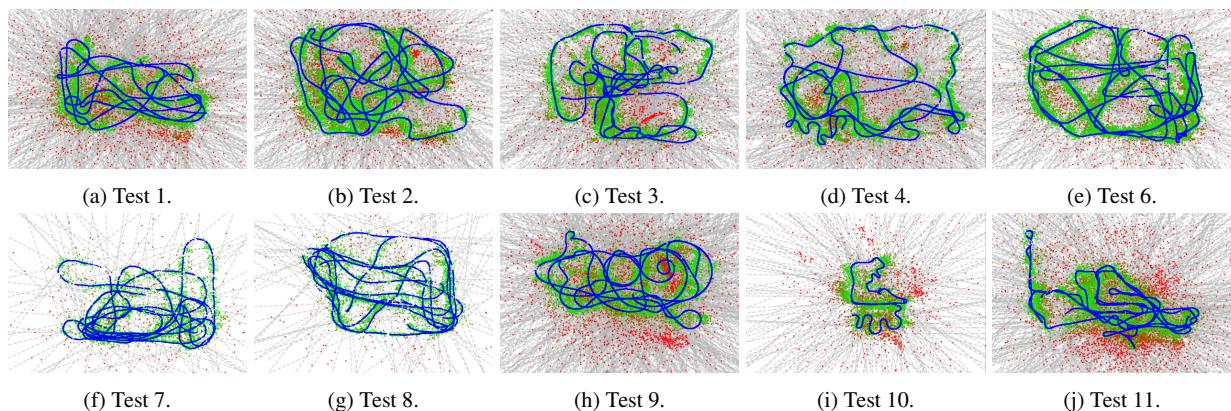


Figure 34: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 4 dataset.

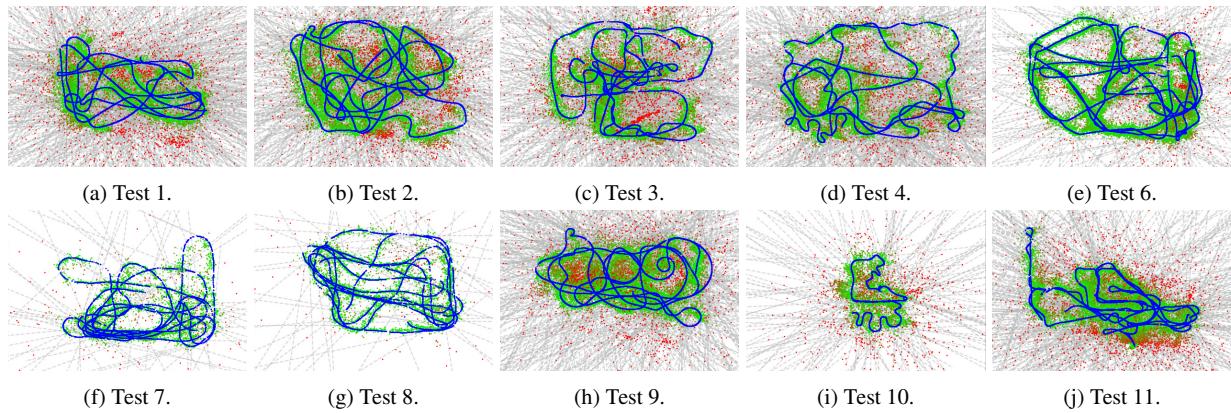


Figure 35: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 5 dataset.

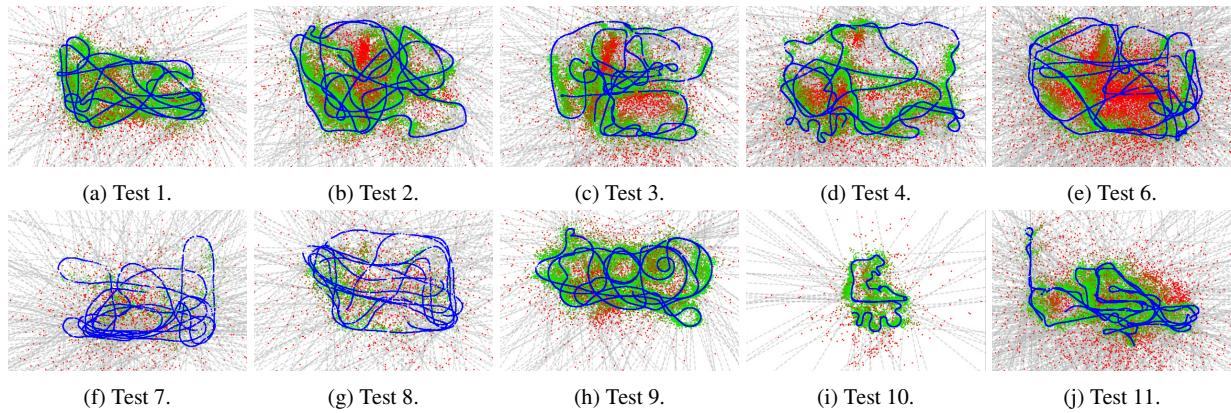


Figure 36: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 6 dataset.

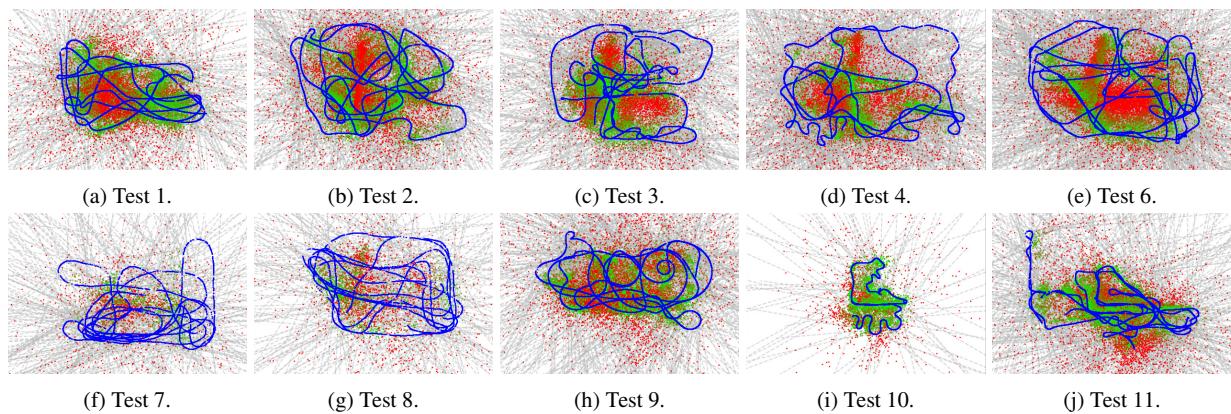


Figure 37: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 7 dataset.

12. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

360

Fusing SfM and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments

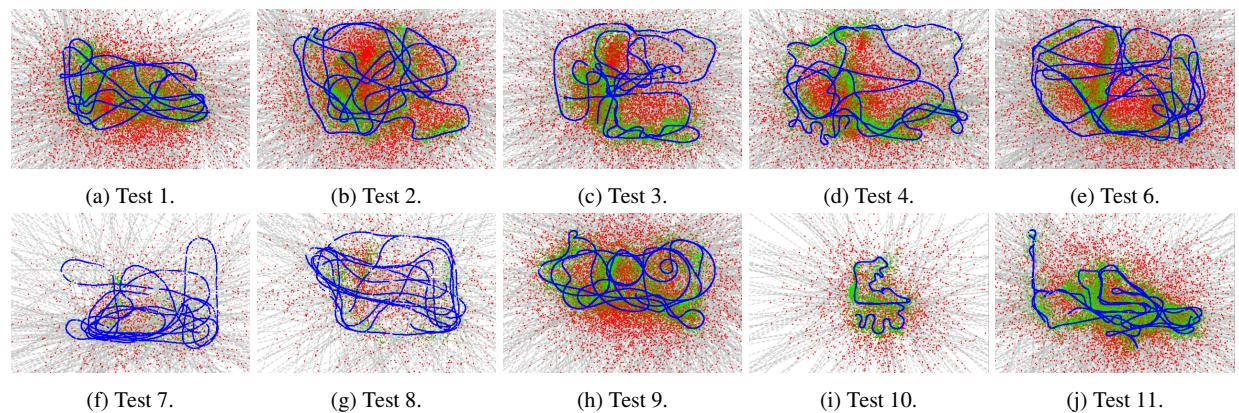


Figure 38: Evaluation of the predicted positions (green, red) against the ground truth trajectories (blue) for SfM for the train 8 dataset.

Biography



Felix Ott received his MSc. degree in Computational Engineering at the FAU Erlangen-Nürnberg in 2019. He joined the Hybrid Positioning & Information Fusion group in the Locating and Communication Systems department at Fraunhofer IIS. In 2020 he started his Ph.D. at the Ludwig-Maximilians University in Munich in the Probabilistic Machine and Deep Learning group. His research covers multimodal information fusion for self-localization.



Lucas Heublein received his M.Sc. degree in Integrated Life Science at the FAU Erlangen-Nürnberg. In 2020, he started his Computer Science degree at the FAU. He joined the Hybrid Positioning & Information Fusion group at the Fraunhofer IIS in 2020 as a student assistant.



David Rügamer is an interim professor for Computational Statistics at the RWTH Aachen. Before he was research associate, lecturer and interim professor for Data Science at the LMU Munich, where he also received his Ph.D. in 2018. His research is concerned with scalability of statistical modeling as well as machine learning for functional and multimodal data.



Bernd Bischl is a full professor for statistical learning and data science and a director of the Munich Center of Machine Learning. His research focuses amongst other things on AutoML, interpretable machine learning and ML benchmarking.



Christopher Mutschler leads the precise positioning and analytics department at Fraunhofer IIS. Prior to that, Christopher headed the Machine Learning & Information Fusion group. He gives lectures on machine learning at the FAU Erlangen-Nürnberg (FAU), from which he also received both his Diploma and PhD in 2010 and 2014 respectively. Christopher's research combines machine learning with radio-based localization.

Contributing Publications

Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 4(3), article 92, pages 1–20, Cancún, Mexico, September 2020. doi:10.1145/3411842. ↗ 109

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In *Proceedings of the IEEE/CVF Winter Conference for Applications on Computer Vision (WACV)*, pages 266–276, Waikoloa, HI, January 2022. doi:10.1109/WACV51458.2022.00131. ↗ 133

Andreas Klaß, Sven M. Lorenz, Martin W. Lauer-Schmaltz, David Rügamer, Bernd Bischl, Christopher Mutschler, and Felix Ott. Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift. In *IJCAI-ECAI International Workshop on Spatio-Temporal Reasoning and Learning (STRL)*, volume 3190, Vienna, Austria, July 2022. ↗ 153

Felix Ott, David Rügamer, Lucas Heublein, Tim Hamann, Jens Barth, Bernd Bischl, and Christopher Mutschler. Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In *International Journal on Document Analysis and Recognition (IJDAR)*, volume 25(12), pages 385–414, September 2022. doi:10.1007/s10032-022-00415-6. ↗ 167

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In *Proceedings of the ACM International Conference on Multimedia (ACMMM)*, pages 5934–5943, Lisboa, Portugal, October 2022. doi:10.1145/3503161.3548167. ↗ 201

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition. In *IEEE Access*, volume 11, pages 94148–94172, August 2023. doi:10.1109/ACCESS.2023.3310819. ↗ 221

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition. In *IAPR International Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS)*, volume 13643, pages 373-383, Montreal, Canada, August 2022. Available at *arXiv preprint arXiv:2301.06293 [cs.CV]*. doi:10.1007/978-3-031-37660-3_26.  265

Felix Ott, Tobias Feigl, Christoffer Löfller, and Christopher Mutschler. ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 187-198, Seattle, WA, June 2020. doi:10.1109/CVPRW50498.2020.00029.  281

Felix Ott, Nisha Lakshmana Raichur, David Rügamer, Tobias Feigl, Heiko Neumann, Bernd Bischl, and Christopher Mutschler. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression. *arXiv preprint arXiv:2208.00919v3 [cs.CV]*, August 2023.  297

Felix Ott, Lucas Heublein, David Rügamer, Bernd Bischl, and Christopher Mutschler. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments. *Submitted to Computer Vision and Image Understanding (CVIU), Elsevier*, Available at *arXiv preprint arXiv:2304.07250v2 [cs.CV]*, July 2023.  329

Further Publications

Maximilian Stahlke, Sebastian Kram, Felix Ott, Tobias Feigl, and Christopher Mutschler. Estimating ToA Reliability with Variational Autoencoders. In *IEEE Sensors Journal*, volume 22(6), pages 5133-5140, March 2022, doi:10.1109/JSEN.2021.3101933.

Nisha Lakshmana Raichur, Tobias Brieger, Dorsaf Jdidi, Tobias Feigl, Johannes Rossouw van der Merwe, Birendra Ghimire, Felix Ott, Alexander Rügamer, and Wolfgang Felber. Machine Learning-assisted GNSS Interference Monitoring Through Crowdsourcing. In *Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+)*, pages 1151-1175, Denver, CO, September 2022, doi:10.33012/2022.18492.

Tobias Brieger, Nisha Lakshmana Raichur, Dorsaf Jdidi, Felix Ott, Tobias Feigl, Johannes Rossouw van der Merwe, Alexander Rügamer, and Wolfgang Felber. Multimodal Learning for Reliable Interference Classification in GNSS Signals. In *Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+)*, pages 3210-3234, Denver, CO, September 2022, doi:10.33012/2022.18586.

Johannes Rossouw van der Merwe, David Contreras Franco, Jonathan Hansen, Tobias Brieger, Tobias Feigl, Felix Ott, Dorsaf Jdidi, Alexander Rügamer, and Wolfgang Felber. Low-Cost COTS GNSS Interference Monitoring, Detection, and Classification System. In *MDPI Sensors*, volume 23(7), 3452, March 2023, doi:10.3390/s23073452.

Bibliography

A. Abanda, U. Mori, and J. A. Lozano. Ad-hoc Explanation for Time Series Classification. In *Knowledge-Based Systems*, volume 252(C), September 2022. doi: 10.1016/j.knosys.2022.109366.

Shailesh Acharya, Ashok Kumar Pant, and Prashnna Kumar Gyawali. Deep Learning based Large Scale Handwritten Devangari Character Recognition. In *Intl. Conf. on Software, Knowledge, Information Management and Applications (SKIMA)*, Kathmandu, Nepal, December 2015. doi: 10.1109/SKIMA.2015.7400041.

Muhammad Shamsul Alam, Farhan Bin Mohamed, Ali Selamat, and AKM Bellal Hossain. Optimizing Indoor Camera Localization: A Novel Approach with Motion Blur Elimination and Recurrent Deep Architecture from Image Sequences. In *SSRN*, June 2023. doi: 10.2139/ssrn.4474209.

Tsige Tadesse Alemayoh, Masaaki Shintani, Jae Hoon Lee, and Shingo Okamoto. Deep-Learning-Based Character Recognition from Handwriting Motion Data Captured Using IMU and Force Sensors. In *MDPI Sensors*, volume 22(20), October 2022. doi: 10.3390/s22207840.

Fevzi Alimoglu and Ethem Alpaydin. Combining Multiple Representations and Classifiers for Pen-based Handwritten Digit Recognition. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, volume 2, Ulm, Germany, August 1997. doi: 10.1109/ICDAR.1997.620583.

Naimeh Alipour and Jafar Tahmoresnezhad. Heterogeneous Domain Adaptation with Statistical Distribution Alignment and Progressive Pseudo Label Selection. In *Applied Intelligence*, volume 52, pp. 8038–8055, October 2021. doi: 10.1007/s10489-021-02756-x.

María A. Pérez Alonso. Metacognition and Sensorimotor Components Underlying the Process of Handwriting and Keyboarding and Their Impact on Learning. An Analysis from the Perspective of Embodied Psychology. In *Procedia – Social and Behavioral Sciences*, volume 176, pp. 263–269, 2015. doi: 10.1016/j.sbspro.2015.01.470.

Mohammad Altillawi. PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pp. 4156–4162, May 2022. doi: 10.1109/ICRA46639.2022.9812345.

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L. Reyes-Ortiz. A Public Domain Dataset for Human Activity Recognition Using Smartphones. In *Proc. of the Europ. Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pp. 437–442, Bruges, Belgium, April 2013.

Hilda Azimi, Steven Chang, Jonathan Gold, and Koray Karabina. Improving Accuracy and Explainability of Online Handwriting Recognition. In *arXiv preprint arXiv:2209.09102v1 [cs.CV]*, September 2022.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. In *arXiv preprint arXiv:1803.01271v1 [cs.LG]*, April 2018.

Simon Baker and Iain Matthews. Lucas-Kanade 20 Years on: A Unifying Framework. In *Intl. Journal of Computer Vision (IJCV)*, volume 56, pp. 221–255, February 2004. doi: 10.1023/B:VISI.0000011205.11775.fd.

David Balduzzi and Muhammad Ghifary. Strongly-Typed Recurrent Neural Networks. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, volume 48, pp. 1292–1300, New York, NY, 2017. doi: 10.5555/3045390.3045527.

Darwin Bautista and Rowel Atienza. Scene Text Recognition with Permuted Autoregressive Sequence Models. In *Europ. Conf. on Computer Vision (ECCV)*, pp. 178–196, October 2022. doi: 10.1007/978-3-031-19815-1_11.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A Theory of Learning From Different Domains. In *Machine Learning*, volume 79, pp. 151–175, October 2009. doi: 10.1007/s10994-009-5152-4.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. In *arXiv preprint arXiv:1206.5538v3 [cs.LG]*, April 2014.

Benjamin Blankertz, Gabriel Curio, and Klaus-Robert Müller. Classifying Single Trial EEG: Towards Brain Computer Interfacing. In *Advances of Neural Information Processing Systems (NIPS)*, volume 14, pp. 157–164, 2001. doi: 10.5555/2980539.2980561.

Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J. Smola. Integrating Structured Biological Data by Kernel Maximum Mean Discrepancy. In *Bioinformatics*, volume 22(14), pp. e49–e57, July 2006. doi: 10.1093/bioinformatics/btl242.

Said Yacine Boulahia, Abdenour Amamra, Mohamed Ridha Madi, and Said Daikh. Early, Intermediate and Late Fusion Strategies for Robust Deep Learning-based Multimodal Action Recognition. In *Machine Vision and Applications*, volume 32(121), September 2021. doi: 10.1007/s00138-021-01249-8.

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. Quasi-Recurrent Neural Networks. In *Intl. Conf. on Learning Representations (ICLR)*, Toulon, France, April 2017.

Hervé Bredin. TristouNet: Triplet Loss for Speaker Turn Embedding. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5430–5434, March 2017. doi: 10.1109/ICASSP.2017.7953194.

Tobias Brieger, Nisha Lakshmana Raichur, Dorsaf Jdidi, Felix Ott, Tobias Feigl, Johannes Rossouw van der Merwe, Alexander Rügamer, and Wolfgang Felber. Multimodal Learning for Reliable Interference Classification in GNSS Signals. In *Proc. of the Intl. Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+)*, pp. 3210–3234, Denver, CO, September 2022. doi: 10.33012/2022.18586.

Matteo Bronkhorst. A Pen is All You Need. In *Twente Student Conf. on IT*, Enschede, The Netherlands, 2021.

Yanling Bu, Lei Xie, Yafeng Yin, Chuyu Wang, Jingyi Ning, Jiannong Cao, and Sanglu Lu. Handwriting-Assistant: Reconstructing Continuous Strokes with Millimeter-level Accuracy via Attachable Inertial Sensors. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 5(4), article 146, pp. 1–25, December 2021. doi: doi/10.1145/3494956.

Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W. Achtelik, and Roland Siegwart. The EuRoC Micro Aerial Vehicle Datasets. In *Intl. Journal of Robotics Research (IJRR)*, volume 35(10), pp. 1157–1163, January 2016. doi: 10.1177/0278364915620033.

Aldrich A. Cabrera-Ponce, Manuel Martin-Ortiz, and J. Martinez-Carranza. Continual Learning for Multi-Camera Relocalisation. In *Mexican Intl. Conf. on Artificial Intelligence (MICAI)*, volume 13067, pp. 389–302, October 2021. doi: 10.1007/978-3-030-89817-5_22.

Jian-Feng Cai, Emmanuel J. Candés, and Zuowei Shen. A Singular Value Thresholding Algorithm for Matrix Completion. In *SIAM Journal on Optimization (SIOPT)*, volume 20(4), pp. 1956–1982, March 2010. doi: 10.1137/080738970.

Fabio M. Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio D. Spano, and Andrea Giachetti. Comparing 3D Trajectories for Simple Mid-Air Gesture Recognition. In *Computers & Graphics*, volume 73, pp. 17–25, June 2018. doi: 10.1016/j.cag.2018.02.009.

Josep L. Carrasco and Lluís Jover. Estimating the Generalized Concordance Correlation Coefficient Through Variance Components. In *Biometrics*, volume 59(4), pp. 849–858, December 2003. doi: 10.1111/j.0006-341x.2003.00099.x.

Stanley H. Chan. Introduction to Probability for Data Science. In *Michigan Publishing Services*, November 2021. ISBN 978-1-60785-746-4.

Kartik Chaudhary and Raghav Bali. Easter2.0: Improving Convolutional Models for Handwritten Text Recognition. In *arXiv preprint arXiv:2205.14879v1 [cs.CV]*, May 2022.

Ushasi Chaudhuri, Biplab Banerjee, Avik Bhattacharya, and Mihai Datcu. CrossATNet – A Novel Cross-Attention based Framework for Sketch-based Image Retrieval. In *Image and Vision Computing*, volume 104, December 2020. doi: 10.1016/j.imavis.2020.104003.

Changhao Chen, Stefano Rosa, Yishu Miao, Chris Xiaoxuan Lu, Wei Wu, Andrew Markham, and Niki Trigoni. Selective Sensor Fusion for Neural Visual-Inertial Odometry. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 10542–10551, Long Beach, CA, June 2019. doi: 10.1109/CVPR.2019.01079.

Chao Chen, Zhihang Fu, Zhihong Chen, Sheng Jin, Zhaowei Cheng, Xinyu Jin, and Xian sheng Hua. HoMM: Higher-Order Moment Matching for Unsupervised Domain Adaptation. In *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, volume 34(4), pp. 3422–3429, April 2020. doi: 10.1609/aaai.v34i04.5745.

Junshan Kevin Chen, Wanze Xie, and Yutong (Kelly) He. Motion-based Handwriting Recognition. In *arXiv preprint arXiv:2101.06022v1 [cs.CV]*, January 2021a.

Shuai Chen, Zirui Wang, and Victor Prisacariu. Direct-PoseNet: Absolute Pose Regression with Photometric Consistency. In *IEEE Intl. Conf. on 3D Vision (3DV)*, London, United Kingdom, December 2021b. doi: 10.1109/3DV53792.2021.00125.

Shuai Chen, Xinghui Li, Zirui Wang, and Victor A. Prisacariu. DFNet: Enhance Absolute Pose Regression with Direct Feature Matching. In *Europ. Conf. on Computer Vision (ECCV)*, November 2022a. doi: 10.1007/978-3-031-20080-9_1.

Wei Chen, Yu Liu, Erwin M. Bakker, and Michael S. Lew. Integrating Information Theory and Adversarial Learning for Cross-Modal Retrieval. In *Pattern Recognition*, volume 117(107983), September 2021c. doi: 10.1016/j.patcog.2021.107983.

Zhounan Chen, Daihui Yang, Jinglin Liang, Xinwu Liu, Yuyi Wang, Zhenghua Peng, and Shuangping Huang. Complex Handwriting Trajectory Recovery: Evaluation Metrics and Algorithm. In *Proc. of the IEEE/CVF Asian Conf. on Computer Vision (ACCV)*, volume 13842, pp. 58–74, February 2022b. doi: 10.1007/978-3-031-26284-5_4.

Xiuyuan Cheng and Yao Xie. Neural Tangent Kernel Maximum Mean Discrepancy. In *Advances in Neural Information Processing Systems (NIPS)*, 2021.

Anoop Cherian, Suvrit Sra, Arindam Banerjee, and Nikolaos Papanikolopoulos. Jensen-Bregman LogDet Divergence with Application to Efficient Similarity Search for Covariance Matrices. In *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 35(9), pp. 2161–2174, December 2012. doi: 10.1109/TPAMI.2012.259.

Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking Attention with Performers. In *Intl. Conf. on Learning Representations (ICLR)*, 2021.

Vincent Christlein, David Bernecker, Andreas Maier, and Elli Angelopoulou. Offline Writer Identification Using Convolutional Neural Network Activation Features. In *DAGM German Conf. on Pattern Recognition*, volume 9358, November 2015. doi: 10.1007/978-3-319-24947-6_45.

Sanghyuk Chun, Seong Joon Oh, Rafael Sampaio de Rezende, Yannis Kalantidis, and Diane Larlus. Probabilistic Embeddings for Cross-Modal Retrieval. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 8415–8424, Nashville, TN, June 2021. doi: 10.1109/CVPR46437.2021.00831.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In *Advances of Neural Information Processing Systems Workshops (NIPSW) on Deep Learning*, December 2014.

Andrzej Cichocki, Sergio Cruces, and Shun-ichi Amari. Log-Determinant Divergences Revisited: Alpha-Beta and Gamma Log-Det Divergences. In *Entropy*, volume 17(5), pp. 2988–3034, May 2015. doi: 10.3390/e17052988.

Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal Transport for Domain Adaptation. In *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 39(9), pp. 1853–1865, October 2016. doi: 10.1109/TPAMI.2016.2615921.

Marco Cuturi. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *Advances of Neural Information Processing Systems (NIPS)*, pp. 2292–2300, December 2013. doi: 10.5555/2999792.2999868.

Fred J. Damerau. A Technique for Computer Detection and Correction of Spelling Errors. In *Communications of the ACM*, volume 7(3), pp. 171–176, March 1964. doi: 10.1145/363958.363994.

Oscar Day and Taghi M. Khoshgoftaar. A Survey on Heterogeneous Transfer Learning. In *Journal of Big Data*, volume 4(29), September 2017. doi: 10.1186/s40537-017-0089-0.

Arthur Flor de Sousa Neto, Byron Leite Dantas Bezerra, Alejandro Hector Toselli, and Estanislau Baptista Lima. A Robust Handwritten Recognition System for Learning on Different Data Restriction Scenarios. In *Pattern Recognition Letters*, volume 159, pp. 232–238, July 2022. doi: 10.1016/j.patrec.2022.04.009.

Namrata Deka and Danica J. Sutherland. MMD-B-Fair: Learning Fair Representations with Statistical Testing. In *Proc. of the Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 206, pp. 9564–9576, Valencia, Spain, April 2023.

Shohreh Deldari, Hao Xue, Aaqib Saeed, Jiayuan He, Daniel V. Smith, and Flora D. Salim. Beyond Just Vision: A Review on Self-Supervised Representation Learning on Multimodal and Temporal Data. In *arXiv preprint arXiv:2206.02353v2 [cs.LG]*, June 2022a.

Shohreh Deldari, Hao Xue, Aaqib Saeed, Daniel V. Smith, and Flora D. Salim. COCOA: Cross Modality Contrastive Learning for Sensor Data. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 6(3), article 108, pp. 1–28, Atlanta, GA & Cambridge, UK, September 2022b. doi: 10.1145/3550316.

Thomas Deselaers, Daniel Keysers, Jan Hosang, and Henry A. Rowley. GyroPen: Gyroscopes for Pen-Input with Mobile Phones. In *Trans. on Human-Machine Systems*, volume 45(2), pp. 263–271, April 2015. doi: 10.1109/THMS.2014.2365723.

Daniel Hernandez Diaz, Siyang Qin, Reeve Ingle, Yasuhisa Fujii, and Alessandro Bisacco. Rethinking Text Line Recognition Models. In *arXiv preprint arXiv:2104.07787v2 [cs.CV]*, April 2021.

Thanh-Toan Do, Toan Tran, Ian Reid, Vijay Kumar, Tuan Hoang, and Gustavo Carneiro. A Theoretically Sound Upper Bound on the Triplet Loss for Improving the Efficiency of Deep Distance Metric Learning. In *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 10404–10413, Long Beach, CA, June 2019. doi: 10.1109/CVPR.2019.01065.

João Paulo Silva do Monte Lima, Hideaki Uchiyama, and Rin ichiro Taniguchi. End-to-End Learning Framework for IMU-based 6-DOF Odometry. In *MDPI Sensors*, volume 19(17), pp. 3777, August 2019. doi: 10.3390/s19173777.

Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philipp Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, pp. 2758–2766, Santiago de Chile, Chile, December 2015. doi: 10.1109/ICCV.2015.316.

Tobias Drey, Jessica Janek, Josef Lang, Dietmar Puschmann, Michael Rietzler, and Enrico Rukzio. SpARKlingPaper: Enhancing Common Pen- and Paper-based Handwriting Training for Children by Digitally Augmenting Papers Using a Tablet Screen. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 6(3), article 113, pp. 1–29, Atlanta, GA & Cambridge, UK, September 2022. doi: 10.1145/3550337.

Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, and Xiaoli Li. Label-efficient Time Series Representation Learning: A Review. In *arXiv preprint arXiv:2302.06433v1 [cs.LG]*, February 2023.

Nelly Elsayed, Anthony S. Maida, and Magdy Bayoumi. Deep Gated Recurrent and Convolutional Network Hybrid Model for Univariate Time Series Classification. In *Intl. Journal of Advanced Computer Science and Applications (IJACSA)*, volume 10(5), 2019. doi: 10.14569/IJACSA.2019.0100582.

Maged M. M. Fahmy. Online Signature Verification and Handwriting Classification. In *Ain Shams Engineering Journal (ASEJ)*, volume 1(1), pp. 59–70, September 2010. doi: 10.1016/j.asej.2010.09.007.

Marcos Faundez-Zanuy and Jiri Mekyska. Analysis of Gender Differences in Online Handwriting Signals for Enhancing e-Health and e-Security Applications. In *Cognitive Computation*, January 2023. doi: 10.1007/s12559-023-10116-9.

Kevin Fauvel, Élisa Fromont, Véronique Masson, Philippe Faverdin, and Alexandre Terrier. XEM: An Explainable-by-Design Ensemble Method for Multivariate Time Series Classification. In *WIREs Data Mining and Knowledge Discovery*, volume 36, pp. 917–957, February 2022. doi: 10.1007/s10618-022-00823-6.

Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weberf, Geoffrey I. Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. InceptionTime: Finding AlexNet for Time Series Classification. In *WIREs Data Mining and Knowledge Discovery*, volume 34, pp. 1936–1962, September 2020. doi: 10.1007/s10618-020-00710-y.

Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouvé, and Gabriel Peyré. Interpolating Between Optimal Transport and MMD Using Sinkhorn Divergences. In *Proc. of the Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 89, pp. 2681–2690, Naha, Okinawa, Japan, April 2019.

John Fitzpatrick. The Writings of Washington from the Original Manuscript Sources, 1745–1799. In *Government Printing Office*, 1931.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T. H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. POT: Python Optimal Transport. In *Journal of Machine Learning Research (JMLR)*, volume 22, pp. 1–8, April 2021. URL <https://pythonot.github.io/index.html>.

Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, and Roee Litman. ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation. In *Proc. of*

the *IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 4324–4333, Seattle, WA, June 2020. doi: 10.1109/CVPR42600.2020.00438.

Ji Gan, Weiqiang Wang, Jiaxu Leng, and Xinbo Gao. HiGAN+: Handwriting Imitation GAN with Disentangled Representations. In *ACM Trans. on Graphics (TOG)*, pp. 1–17, February 2023. doi: 10.1145/3550070.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. In *Journal of Machine Learning Research (JMLR)*, volume 17(59), pp. 1–35, 2016.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision Meets Robotics: The KITTI Dataset. In *Intl. Journal of Robotics Research (IJRR)*, volume 32(11), pp. 1231–1237, August 2013. doi: 10.1177/0278364913491297.

Aude Genevay, Gabriel Peyre, and Marco Cuturi. Learning Generative Models with Sinkhorn Divergences. In *Proc. of the Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 84, pp. 1608–1617, 2018.

Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Spectral, Probabilistic, and Deep Metric Learning: Tutorial and Survey. In *arXiv preprint arXiv:2201.09267v1 [stat.ML]*, January 2022.

Trishita Ghosh, Shibaprasad Sen, Sk.Md. Obaidullah, K.C. Santosh, Kaushik Roy, and Umapada Pal. Advances in Online Handwritten Recognition in the Last Decades. In *Computer Science Review*, volume 46, November 2022. doi: 10.1016/j.cosrev.2022.100515.

Graham L. Giller. The Statistical Properties of Random Bitstreams and the Sampling Distribution of Cosine Similarity. In *SSRN*, October 2012. doi: 10.2139/ssrn.2167044.

Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. In *Circulation*, volume 101(23), pp. 215–220, June 2000. doi: 10.1161/01.cir.101.23.e215.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. In *MIT Press*, 2016.

Mononito Goswami, Christian Ignacio Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. Unsupervised Model Selection for Time Series Anomaly Detection. In *Intl. Conf. on Learning Representations (ICLR)*, May 2023.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent

Neural Networks. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, pp. 369–376, Pittsburgh, PA, June 2006. doi: 10.1145/1143844.1143891.

Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A Novel Connectionist System for Unconstrained Handwriting Recognition. In *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 31(5), pp. 855–868, May 2009. doi: 10.1109/TPAMI.2008.137.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A Kernel Two-Sample Test. In *Journal of Machine Learning Research (JMLR)*, volume 13, pp. 723–773, March 2012. doi: 10.5555/2188385.2188410.

Matthew Koichi Grimes, Dragomir Anguelov, and Yann LeCun. Hybrid Hessians for Flexible Optimization of Pose Graphs. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 2010. doi: 10.1109/IROS.2010.5650091.

Xiangming Gu, Longshen Ou, Danielle Ong, and Ye Wang. MM-ALT: A Multimodal Automatic Lyric Transcription System. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*, pp. 3328–3337, Lisboa, Portugal, October 2022. doi: 10.1145/3503161.3548411.

Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel. AudioCLIP: Extending Clip to Image, Text and Audio. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 976–980, Singapore, Singapore, May 2022. doi: 10.1109/ICASSP43922.2022.9747631.

Mehrtash Harandi, Mathieu Salzmann, and Fatih Porikli. Bregman Divergences for Infinite Dimensional Covariance Matrices. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1003–1010, Columbus, OH, June 2014. doi: 10.1109/CVPR.2014.132.

Guozheng He, Zhouyi Wu, Yuting Wu, Peiying Lin, and Hiantao Huangfu. Online Handwriting Recognition Based on Microphone and IMU. In *Intl. Conf. on Electronics Technology (ICET)*, Chengdu, China, May 2022a. doi: 10.1109/ICET55676.2022.9824489.

Huan He, Owen Queen, Teddy Koker, Consuelo Cuevas, Theodoros Tsiligkaridis, and Marinka Zitnik. Domain Adaptation for Time Series Under Feature and Label Shifts. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, July 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, June 2016. doi: 10.1109/CVPR.2016.90.

Qiang He, Zhiping Feng, Xue Wang, Yufen Wu, and Jin Wang. A Smart Pen Based on Triboelectric Effects for Handwriting Pattern Tracking and Biometric Identification. In

ACS Appl. Mater. Interfaces, volume 14(43), pp. 49295–49302, October 2022b. doi: 10.1021/acsami.2c13714.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of Tricks for Image Classification with Convolutional Neural Networks. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 558–567, Long Beach, CA, June 2019. doi: 10.1109/CVPR.2019.00065.

Geoffrey E. Hinton and Sam Roweis. Stochastic Neighbor Embedding. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. In *Neural Computation*, volume 9(8), pp. 1735–1780, 1997.

Xin Huang, Yuxin Peng, and Mingkuan Yuan. MHTN: Modal-Adversarial Hybrid Transfer Network for Cross-Modal Retrieval. In *Trans. on Cybernetics*, volume 50(3), pp. 1047–1059, March 2020. doi: 10.1109/TCYB.2018.2879846.

Peter J. Huber and Elvezio M. Ronchetti. Robust Statistics, Second Edition. In *Wiley Series in Probability and Statistics*, January 2009. doi: 10.1002/9780470434697.

Raashid Hussain, Ahsen Raza, Imran Siddiqi, Khurram Khurshid, and Chawki Djeddi. A Comprehensive Survey of Handwritten Document Benchmarks: Structure, Usage and Evaluation. In *EURASIP Journal on Image and Video Processing*, volume 46, 2015. doi: 10.1186/s13640-015-0102-5.

Eyke Hüllermeier, Sébastien Destercke, and Mohammad Hossein Shaker. Quantification of Credal Uncertainty in Machine Learning: A Critical Analysis and Empirical Comparison. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, volume 180, pp. 548–557, August 2022.

Aya S. Ihara, Kae Nakajima, Akiyuki Kake, Kizuku Ishimaru, Kiyoyuki Osugi, and Yasushi Naruse. Advantage of Handwriting Over Typing on Learning Words: Evidence From an N400 Event-Related Potential Index. In *Front. Hum. Neurosci.*, volume 15, June 2021. doi: 10.3389/fnhum.2021.679191.

Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1647–1655, Honolulu, HI, July 2017. doi: 10.1109/CVPR.2017.179.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, volume 37, pp. 448–456, July 2015. doi: 10.5555/3045118.3045167.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General Perception with Iterative Attention. In *Journal of Machine Learning Research (JMLR)*, volume 130, pp. 4651–4664, 2021.

Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. ColloSSL: Collaborative Self-Supervised Learning for Human Activity Recognition. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 6(1), article 17, pp. 1–28, Atlanta, GA & Cambridge, UK, March 2022. doi: 10.1145/3517246.

Monica Jangpangi, Sudhanshu Kumar, Diwakar Bhardwaj, Byung-Gyu Kim, and Partha Pratim Roy. Handwriting Recognition Using Wasserstein Metric in Adversarial Learning. In *SN Computer Science*, volume 4(43), November 2022. doi: 10.1007/s42979-022-01445-x.

Junguang Jiang, Ximei Wang, Mingsheng Long, and Jianmin Wang. Resource Efficient Domain Adaptation. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*, pp. 2220–2228, Seattle, WA, October 2020a. doi: 10.1145/3394171.3413701.

San Jiang, Cheng Jiang, and Wanshou Jiang. Efficient Structure from Motion for Large-scale UAV Images: A Review and a Comparison of SfM Tools. In *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*, volume 167, pp. 230–251, September 2020b. doi: 10.1016/j.isprsjprs.2020.04.016.

Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-On Bayesian Neural Networks – A Tutorial for Deep Learning Users. In *IEEE Computational Intelligence Magazine (CIM)*, volume 17(2), pp. 29–48, April 2022. doi: 10.1109/MCI.2022.3155327.

Hamid Reza Vaezi Joze, Amirreza Shaban, Michael L. Iuzzolino, and Kazuhito Koishida. MMTM: Multimodal Transfer Module for CNN Fusion. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 13289–13299, Seattle, WA, June 2020. doi: 10.1109/CVPR42600.2020.01330.

Eugenio Peixoto Júnior, Italo L. D. Delmiro, Naercio Magaia, Fernanda M. Maia, Mohammad Mehedi Hassan, Victor Hugo C. Albuquerque, and Giancarlo Fortino. Intelligent Sensory Pen for Aiding in the Diagnosis of Parkinson’s Disease from Dynamic Handwriting Analysis. In *MDPI Sensors*, volume 20(20), October 2020. doi: 10.3390/s20205840.

Lei Kang, Pau Riba, Marcal Rusinol, Alicia Fornés, and Mauricio Villegas. Content and Style Aware Generation of Text-line Images for Handwriting Recognition. In *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 44(12), pp. 8846–8860, October 2021. doi: 10.1109/TPAMI.2021.3122572.

Lei Kang, Pau Riba, Marcal Rusinol, Alicia Fornes, and Mauricio Villegas. Pay Attention to What You Read: Non-recurrent Handwritten Text-Line Recognition. In *Pattern Recognition*, volume 129(C), September 2022. doi: 10.1016/j.patcog.2022.108766.

L. V. Kantorovitch. On the Translocation of Masses. In *Journal of Mathematical Sciences*, volume 133, pp. 1381–1382, March 2006. doi: 10.1007/s10958-006-0049-2.

Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. LSTM Fully Convolutional Networks for Time Series Classification. In *IEEE Access*, volume 6, pp. 1662–1669, December 2017. doi: 10.1109/ACCESS.2017.2779939.

Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate LSTM-FCNs for Time Series Classification. In *Neural Network*, volume 116, pp. 237–245, August 2019. doi: 10.1016/j.neunet.2019.04.014.

Alex Kendall and Yarin Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances of Neural Information Processing Systems (NIPS)*, pp. 5580–5590, December 2017. doi: 10.5555/3295222.3295309.

Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, pp. 2938–2946, Santiago de Chile, Chile, December 2015. doi: 10.1109/ICCV.2015.336.

Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting Change in Data Streams. In *Proc. of the Intl. Conf. on Very Large Data Bases (VLDB)*, volume 30, pp. 180–191, August 2004. doi: 10.5555/1316689.1316707.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *arXiv preprint arXiv:1412.6980v1 [cs.LG]*, December 2014.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer. In *Intl. Conf. on Learning Representations (ICLR)*, 2020.

Andreas Klaß, Sven M. Lorenz, Martin W. Lauer-Schmaltz, David Rügamer, Bernd Bischl, Christopher Mutschler, and Felix Ott. Uncertainty-aware Evaluation of Time-Series Classification for Online Handwriting Recognition with Domain Shift. In *IJCAI-ECAI Intl. Workshop on Spatio-Temporal Reasoning and Learning (STRRL)*, volume 3190, Vienna, Austria, July 2022. 153.

Jan Kohút and Michal Hradiš. Fine-Tuning is a Surprisingly Effective Domain Adaptation Baseline in Handwriting Recognition. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, volume 14190, pp. 269–286, August 2023. doi: 10.1007/978-3-031-41685-9_17.

Jan Kohút, Michal Hradiš, and Martin Kišš. Towards Writing Style Adaptation in Handwriting Recognition. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, volume 14190, pp. 377–394, August 2023. doi: 10.1007/978-3-031-41685-9_24.

Fabian Kreß, Alexey Serdyuk, Tim Hotfilter, Julian Hoefer, Tanja Harbaum, Jürgen Becker, and Tim Hamann. Hardware-aware Workload Distribution for AI-based Online Handwriting Recognition in a Sensor Pen. In *Mediterranean Conf. on Embedded Computing (MECO)*, Budva, Montenegro, June 2022. doi: 10.1109/MECO55406.2022.9797131.

S. Kullback and R. A. Leibler. On Information and Sufficiency. In *Ann. Math. Statist.*, volume 22(1), pp. 79–86, March 1951. doi: 10.1214/aoms/1177729694.

Kaushal Kumar and Rajib Ghosh. Parkinson’s Disease Diagnosis Using Recurrent Neural Network based Deep Learning Model by Analyzing Online Handwriting. In *Multimedia Tools and Applications*, June 2023. doi: 10.1007/s11042-023-15811-1.

Akisue Kuramoto, Kazuki Hiranai, and Akihiko Seo. Strategies of Pen Tip Path Estimation and of Workload Comparison for Handwriting Tasks. In *IEEE Sensors Journal*, volume 21(3), pp. 3645–3653, October 2020. doi: 10.1109/JSEN.2020.3028605.

Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity Recognition Using Cell Phone Accelerometers. In *ACM SIGKDD Explorations Newsletter*, volume 12(2), pp. 74–82, December 2010. doi: 10.1145/1964897.1964918.

Baptiste Lafabregue, Jonathan Weber, Pierre Gançarski, and Germain Forestier. End-to-End Deep Representation Learning for Time Series Clustering: A Comparative Study. In *WIREs Data Mining and Knowledge Discovery*, volume 36, pp. 29–81, October 2021. doi: 10.1007/s10618-021-00796-y.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 6405–6416, December 2017. doi: 10.5555/3295222.3295387.

Thomas Laurent and James von Brecht. A Recurrent Neural Network Without Chaos. In *Intl. Conf. on Learning Representations (ICLR)*, Toulon, France, April 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. In *Proc. of the IEEE*, volume 86(11), pp. 2278–2324, November 1998.

Kenton Lee, Omer Levy, and Luke Zettlemoyer. Recurrent Additive Networks. In *arXiv preprint arXiv:1705.07393.v2 [cs.CL]*, June 2017.

Sumin Lee, Sangmin Woo, Yeonju Park, Muhammad Adi Nugroho, and Changick Kim. Modality Mixer for Multi-modal Action Recognition. In *Proc. of the IEEE/CVF Winter Conf. for Applications on Computer Vision (WACV)*, Waikoloa, HI, January 2023. doi: 10.1109/WACV56688.2023.00331.

Tao Lei, Yu Zhang, and Yoav Artzi. Training RNNs as Fast as CNNs. In *arXiv preprint arXiv:1709.02755v3 [cs.CL]*, November 2017.

Christian Léonard. A Survey of the Schrödinger Problem and Some of its Connections with Optimal Transport. In *Discrete and Continuous Dynamical Systems*, volume 34(4), pp. 1533–1574, March 2014. doi: 10.3934/dcds.2014.34.1533.

Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models. In *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, February 2023.

Taolin Liao, Tianhang Tang, Jie Chen, Yuting Liang, and Yuhan Xiao. FINet: Feature Interactions Across Dimensions and Hierarchies for Camera Localization. In *Intl. Conf. on Computing and Pattern Recognition (ICCP)*, pp. 43–50, October 2021. doi: 10.1145/3497623.3497631.

Julian Lienen and Eyke Hüllermeier. From Label Smoothing to Label Relaxation. In *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, volume 35(10), pp. 8583–8591, May 2021. doi: 10.1609/aaai.v35i10.17041.

Kaiyi Lin, Xing Xu, Lianli Gao, Zheng Wang, and Heng Tao Shen. Learning Cross-Aligned Latent Embeddings for Zero-Shot Cross-Modal Retrieval. In *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, volume 34(07), pp. 11515–11522, New York, NY, April 2020. doi: 10.1609/aaai.v34i07.6817.

Lawrence I-Kuei Lin. A Concordance Correlation Coefficient to Evaluate Reproducibility. In *Biometrics*, volume 45(1), pp. 255–268, March 1989. doi: 10.2307/2532051.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, pp. 2980–2988, Venice, Italy, 2017. doi: 10.1109/ICCV.2017.324.

Hanxiao Liu, Zihang Dai, David R. So, and Quoc V. Le. Pay Attention to MLPs. In *Advances in Neural Information Processing Systems (NIPS)*, June 2021.

Jiayang Liu, Zhen Wang, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uWave: Accelerometer-based Personalized Gesture Recognition and Its Applications. In *IEEE Intl. Conf. on Pervasive Computing and Communications (PerCom)*, Galveston, TX, March 2009. doi: 10.1109/PERCOM.2009.4912759.

Qiao Liu and Hui Xe. Adversarial Spectral Kernel Matching for Unsupervised Time Series Domain Adaptation. In *Proc. of the Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pp. 2744–2750, August 2021. doi: 10.24963/ijcai.2021/378.

Tianyi Liu and Yusuke Sugano. Interactive Machine Learning on Edge Devices with User-in-the-Loop Sample Recommendation. In *IEEE Access*, volume 10, pp. 107346–107360, October 2022. doi: 10.1109/ACCESS.2022.3212077.

Zhuang Liu, Yunpu Ma, Matthias Schubert, Yuanxin Ouyang, and Zhang Xiong. Multi-Modal Contrastive Pre-Training for Recommendation. In *Proc. of the Intl. Conf. on Multimedia Retrieval (ICMR)*, pp. 99–108, June 2022. doi: 10.1145/3512527.3531378.

Marcus Liwicki and Horst Bunke. IAM-OnDB – an On-Line English Sentence Database Acquired from Handwritten Text on a Whiteboard. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, pp. 956–961, Seoul, Korea, August 2005. doi: 10.1109/ICDAR.2005.132.

David Llorens, Federico Prat, Andrés Marzal, Juan Miguel Vilar, María José Castro-Bleda, Juan Carlos Amengual, Sergio Barrachina Mir, Antonio Castellanos, Salvador España Boquera, Jon Ander Gómez, Jorge Gorbey-Moya, Albert Gordo, Vicente Palazón-González, Guillermo Peris Ripollés, Rafael Ramos-Garijo, and Francisco Zamora-Martínez. The UJIpenchars Database: a Pen-Based Database of Isolated Handwritten Characters. In *Proc. of the Intl. Conf. on Language Resources and Evaluation (LREC)*, Marrakech, Morocco, May 2008.

Christoffer Löffler, Sascha Riechel, Janina Fischer, and Christopher Mutschler. Evaluation Criteria for Inside-Out Indoor Positioning Systems Based on Machine Learning. In *IEEE Intl. Conf. on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–8, Nantes, France, September 2018. doi: 10.1109/IPIN.2018.8533862.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning Transferable Features with Deep Adaptation Networks. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, volume 37, pp. 97–105, July 2015. doi: 10.5555/3045118.3045130.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep Transfer Learning with Joint Adaptation Networks. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, volume 70, pp. 2208–2217, August 2017. doi: 10.5555/3305890.3305909.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional Adversarial Domain Adaptation. In *Advances of Neural Information Processing Systems (NIPS)*, volume 31, pp. 1647–1657, December 2018. doi: 10.5555/3326943.3327094.

Teressa Longjam, Dakshina Ranjan Kisku, and Phalguni Gupta. Writer Independent Handwritten Signature Verification on Multi-Scripted Signatures Using Hybrid CNN-BiLSTM: A Novel Approach. In *Expert Systems with Applications*, volume 214, March 2023. doi: 10.1016/j.eswa.2022.119111.

Canjie Luo, Yuanzhi Zhu, Lianwen Jin, Zhe Li, and Dezhi Peng. SLOGAN: Handwriting Style Synthesis for Arbitrary-Length and Out-of-Vocabulary Text. In *IEEE Trans. on Neural Networks and Learning Systems (TNNLS)*, pp. 1–13, February 2022. doi: 10.1109/TNNLS.2022.3151477.

Pengyuan Lyu, Chengquan Zhang, Shanshan Liu, Meina Qiao, Yangliu Xu, Liang Wu, Kun Yao, Junyu Han, Errui Ding, and Jingdong Wang. MaskOCR: Text Recognition with Masked Encoder-Decoder Pretraining. In *arXiv preprint arXiv:2206.00311v1 [cs.CV]*, June 2022.

Shumin Ma, Zhiri Yuan, Qi Wu, Yiyan Huang, Xixu Hu, Cheuk Hang Leung, Dongdong Wang, and Zhixiang Huang. Deep into the Domain Shift: Transfer Learning Through Dependence Regularization. In *SSRN 4206319*, September 2022. doi: 10.2139/ssrn.4206319.

Wesley J. Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson. A Simple Baseline for Bayesian Uncertainty in Deep Learning. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 13153–13164, December 2019. doi: 10.5555/3454287.3455466.

U.-V. Marti and H. Bunke. A Full English Sentence Database for Offline Handwriting Recognition. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, Bangalore, India, September 1999. doi: 10.1109/ICDAR.1999.791885.

Alexander Mattick, Martin Mayr, Mathias Seuret, Andreas Maier, and Vincent Christlein. SmartPatch: Improving Handwritten Word Imitation with Patch Discriminators. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, pp. 268–283, Lausanne, Switzerland, September 2021. doi: 10.1007/978-3-030-86549-8_18.

Pietro Morerio and Vittorio Murino. Correlation Alignment by Riemannian Metric for Domain Adaptation. In *arXiv preprint arXiv:1705.08180v1 [cs.CV]*, May 2017.

Mohamed Adel Musallam, Vincent Gaudillière, Miguel Ortiz Del Castillo, Kassem Al Ismaeil, and Djamila Aouada. Leveraging Equivariant Features for Absolute Pose Regression. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 6876–6886, New Orleans, LA, June 2022. doi: 10.1109/CVPR52688.2022.00675.

Ahmad Mustafid, Junaid Younas, Paul Lukowicz, and Sheraz Ahmed. IAMonSense: Multi-Level Handwriting Classification Using Spatio-Temporal Information. In *Research Square*, November 2022. doi: 10.21203/rs.3.rs-2275927/v1.

Aviv Navon, Idan Achituve, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary Learning by Implicit Differentiation. In *Intl. Conf. on Learning Representations (ICLR)*, May 2021.

Hung Tuan Nguyen, Cuong Tuan Nguyen, and Masaki Nakagawa. ICFHR 2018 – Competition on Vietnamese Online Handwritten Text Recognition using HANDS-VNOnDB (VOHTR2018). In *IAPR Intl. Conf. on Frontiers in Handwriting Recognition (ICFHR)*, pp. 494–499, Niagara Falls, NY, February 2018. doi: 10.1109/ICFHR-2018.2018.00092.

Ignacio Oguiza. tsai – A State-of-the-art Deep Learning Library for Time Series and Sequential Data. Github, 2020. URL <https://github.com/timeseriesAI/tsai>.

Yasunori Ohishi, Marc Delcroix, Tsubasa Ochiai, Shoko Araki, Daiki Takeuchi, Daisuke Niizumi, Akisato Kimura, Noboru Harada, and Kunio Kashino. ConceptBeam: Concept Driven Target Speech Extraction. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*, pp. 4252–4260, Lisbon, Portugal, October 2022. doi: 10.1145/3503161.3548397.

Emanuele Olivetti, Seved Mostafa Kia, and Paolo Avesani. MEG Decoding Across Subjects. In *Intl. Workshop on Pattern Recognition in Neuroimaging*, Tuebingen, Germany, June 2014. doi: 10.1109/PRNI.2014.6858538.

Felix Ott. Fusing Relative and Absolute Techniques for 6-DoF Visual Self-Localization. In *Master's Thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg*, Erlangen, Germany, January 2019.

Felix Ott, Tobias Feigl, Christoffer Löfller, and Christopher Mutschler. ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 187–198, Seattle, WA, June 2020a. 281. doi: 10.1109/CVPRW50498.2020.00029.

Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. In *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, volume 4(3), article 92, pp. 1–20, Cancún, Mexico, September 2020b. 109. doi: 10.1145/3411842.

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Domain Adaptation for Time-Series Classification to Mitigate Covariate Shift. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*, pp. 5934–5943, Lisboa, Portugal, October 2022a. 201. doi: 10.1145/3503161.3548167.

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Representation Learning for Tablet and Paper Domain Adaptation in Favor of Online Handwriting Recognition. In *IAPR Intl. Workshop on Multimodal Pattern Recognition of Social Signals in Human Computer Interaction (MPRSS)*, volume 13643, pp. 373–383, Montreal, Canada, August 2022b. 265. doi: 10.1007/978-3-031-37660-3_26.

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Joint Classification and Trajectory Regression of Online Handwriting using a Multi-Task Learning Approach. In *Proc. of the IEEE/CVF Winter Conf. for Applications on Computer Vision (WACV)*, pp. 266–276, Waikoloa, HI, January 2022c. 133. doi: 10.1109/WACV51458.2022.00131.

Felix Ott, David Rügamer, Lucas Heublein, Tim Hamann, Jens Barth, Bernd Bischl, and Christopher Mutschler. Benchmarking Online Sequence-to-Sequence and Character-based Handwriting Recognition from IMU-Enhanced Pens. In *International Journal on Document Analysis and Recognition (IJDAR)*, volume 25(12), pp. 385–414. 167, September 2022d. doi: 10.1007/s10032-022-00415-6.

Felix Ott, Lucas Heublein, David Rügamer, Bernd Bischl, and Christopher Mutschler. Fusing Structure from Motion and Simulation-Augmented Pose Regression from Optical Flow for Challenging Indoor Environments. In *arXiv preprint arXiv:2304.07250v2 [cs.CV]*. 329, July 2023a.

Felix Ott, Nisha Lakshmana Raichur, David Rügamer, Tobias Feigl, Heiko Neumann, Bernd Bischl, and Christopher Mutschler. Benchmarking Visual-Inertial Deep Multimodal Fusion for Relative Pose Regression and Odometry-aided Absolute Pose Regression. In *arXiv preprint arXiv:2208.00919v3 [cs.CV]*. 297, August 2023b. doi: 10.48550/arXiv.2208.00919.

Felix Ott, David Rügamer, Lucas Heublein, Bernd Bischl, and Christopher Mutschler. Auxiliary Cross-Modal Representation Learning with Triplet Loss Functions for Online Handwriting Recognition. In *IEEE Access*, volume 11, pp. 94148–94172. 221, August 2023c. doi: 10.1109/ACCESS.2023.3310819.

Yilmazcan Ozyurt, Stefan Feuerriegel, and Ce Zhang. Contrastive Learning for Unsupervised Domain Adaptation for Time Series. In *Intl. Conf. on Learning Representations (ICLR)*, February 2023.

Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. In *IEEE Trans. on Knowledge and Data Engineering*, volume 22(10), pp. 1345–1359, October 2009. doi: 10.1109/TKDE.2009.191.

Vishal M. Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Visual Domain Adaptation: A Survey of Recent Advances. In *IEEE Signal Processing Magazine*, volume 32(3), pp. 53–69, May 2015. doi: 10.1109/MSP.2014.2347059.

Karl Pearson. Notes on the History of Correlation. In *Biometrics*, volume 13(1), October 1920. doi: 10.2307/2331722.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing Neural Networks by Penalizing Confident Output Distributions. In *Intl. Conf. on Learning Representations (ICLR) Workshops*, 2017.

Bernd Pfrommer, Nitin Sanket, Kostas Daniilidis, and Jonas Cleveland. PennCOSYVIO: A Challenging Visual Inertial Odometry Benchmark. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pp. 3847–3854, Singapore, Singapore, May 2017. doi: 10.1109/ICRA.2017.7989443.

Rejean Plamondon and Sargur N. Srihari. On-line and Off-line Handwriting Recognition: A Comprehensive Survey. In *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, volume 22(1), pp. 63–84, January 2000. doi: 10.1109/34.824821.

Noha Radwan, Abhinav Valada, and Wolfram Burgard. VLocNet++: Deep Multi-task Learning for Semantic Visual Localization and Odometry. In *IEEE Robotics and Automation Letters (RA-L)*, volume 3(4), pp. 4407–4414, September 2018. doi: 10.1109/LRA.2018.2869640.

Mohamed Ragab, Emadeldeen Eldele, Wee Ling Tan, Chuan-Sheng Foo, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, and Xiaoli Li. AdaTime: A Benchmarking Suite for Domain Adaptation on Time Series Data. In *Proc. of the ACM Trans. on Knowledge Discovery from Data (TKDD)*, March 2023. doi: 10.1145/3587937.

Elahe Rahimian, Soheil Zabihi, Seyed Farokh Atashzar, Amir Asif, and Arash Mohammadi. XceptionTime: A Novel Deep Architecture based on Depthwise Separable Convolutions for Hand Gesture Classification. In *arXiv preprint arXiv:1911.03803v1 [cs.LG]*, November 2019.

Mohammad Mahfujur Rahman, Clinton Fookes, Mahsa Baktashmotagh, and Sridha Sridharan. On Minimum Discrepancy Estimation for Deep Domain Adaptation. In *Domain Adaptation for Visual Understanding*, Springer, Cham., January 2020. doi: 10.1007/978-3-030-30671-7_6.

Nisha Lakshmana Raichur, Tobias Brieger, Dorsaf Jdidi, Tobias Feigl, Johannes Rossouw van der Merwe, Birendra Ghimire, Felix Ott, Alexander Rügamer, and Wolfgang Felber. Machine Learning-assisted GNSS Interference Monitoring Through Crowdsourcing. In *Proc. of the Intl. Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+)*, pp. 1151–1175, Denver, CO, September 2022. doi: 10.33012/2022.18492.

Viresh Ranjan, Nikhil Rasiwasia, and C. V. Jawahar. Multi-Label Cross-Modal Retrieval. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, pp. 4094–4102, Santiago de Chile, Chile, December 2015. doi: 10.1109/ICCV.2015.466.

Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training Deep Neural Networks on Noisy Labels with Bootstrapping. In *Intl. Conf. on Learning Representations (ICLR) Workshops*, 2015.

Ney Renau-Ferrer, Peiyu Li, Adrien Delaye, and Eric Anquetil. The ILGDB Dataset of Realistic Pen-based Gestural Commands. In *Proc. of the IAPR Intl. Conf. on Pattern Recognition (ICPR)*, Tsukuba, Japan, November 2012.

B. Resch, H. P. A. Lensch, O. Wang, M. Pollefeys, and A. Sorkine-Hornung. Scalable Structure from Motion for Densely Sampled Videos. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 3936–3944, Boston, MA, June 2015. doi: 10.1109/CVPR.2015.7299019.

Mina Rezaei, Emilio Dorigatti, David Rügamer, and Bernd Bischl. Joint Debiased Representation Learning and Imbalanced Data Clustering. In *IEEE Intl. Conf. on Data Mining Workshops (ICDMW)*, pp. 55–62, 2022. doi: 10.1109/ICDMW58026.2022.00016.

Filippo Santambrogio. Optimal Transport for Applied Mathematicians – Calculus of Variations, PDEs, and Modeling. In *Progress in Nonlinear Differential Equations and Their Applications (PNLDE)*, volume 87, 2015. doi: 10.1007/978-3-319-20828-2.

Konstantin Sarin, Marina Bardanova, Mikhail Svetlakov, Nikolay Koryshev, Roman Ostapenko, Antonia Hodashinskaya, and Ilya Hodashinsky. A Three-stage Fuzzy Classifier Method for Parkinson’s Disease Diagnosis Using Dynamic Handwriting Analysis. In *Decision Analytics Journal*, volume 8(100274), September 2023. doi: 10.1016/j.dajour.2023.100274.

Lambert Schomaker. The ICDAR 2003 Informal Competition for the Recognition of On-line Words: The Unipen-ICROW-03 Benchmark Set. In www.ai.rug.nl/lambert/unipen/icdar-03-competition/, 2003.

Maximilian Schrapel, Max-Ludwig Stadler, and Michael Rohs. Pentelligence: Combining Pen Tip Motion and Writing Sounds for Handwritten Digit Recognition. In *Proc. of the ACM Conf. on Human Factors in Computing Systems (CHI)*, pp. 1–11, April 2018. doi: 10.1145/3173574.3173705].

Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823, Boston, MA, June 2015. doi: 10.1109/CVPR.2015.7298682.

Ling Shao, Fan Zhu, and Xuelong Li. Transfer Learning for Visual Categorization: A Survey. In *IEEE Trans. on Neural Networks and Learning Systems (TNNLS)*, volume 26(5), pp. 1019–1034, July 2014. doi: 10.1109/TNNLS.2014.2330900.

Yoli Shavit, Ron Ferens, and Yosi Keller. Learning Multi-Scene Absolute Pose Regression with Transformers. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, Montreal, QC, October 2021. doi: 10.1109/ICCV48922.2021.00273.

Yongjie Shi, Xianghua Ying, and Jinfa Yang. Deep Unsupervised Domain Adaptation with Time Series Sensor Data: A Survey. In *MDPI Sensors*, volume 22(15), July 2022. doi: 10.3390/s22155507.

Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2930–2937, Portland, OR, June 2013. doi: 10.1109/CVPR.2013.377.

Rui Shu, Hung H. Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-T Approach to Unsupervised Domain Adaptation. In *Intl. Conf. on Learning Representations (ICLR)*, 2018.

Damien Simonnet, Nathalie Girard, Eric Anquetil, Mickaël Renault, and Sébastien Thomas. Evaluation of Children Cursive Handwritten Words for e-Education. In *Pattern Recognition Letters*, volume 121, pp. 133–139, 2018.

Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. FLAVA: A Foundational Language and Vision Alignment Model. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 15638–15650, New Orleans, LA, June 2022. doi: 10.1109/CVPR52688.2022.01519.

Shashank Kumar Singh and Amrita Chaturvedi. Leveraging Deep Feature Learning for Wearable Sensors Based Handwritten Character Recognition. In *Biomedical Signal Processing and Control*, volume 80(1), February 2023. doi: 10.1016/j.bspc.2022.104198.

Peeyush Singhal, Rahee Walambe, Sheela Ramanna, and Ketan Kotecha. Domain Adaptation: Challenges, Methods, Datasets, and Applications. In *IEEE Access*, January 2023. doi: 10.1109/ACCESS.2023.3237025.

Suvrit Sra. A New Metric on the Manifold of Kernel Matrices with Application to Matrix Geometric Means. In *Advances in Neural Information Processing Systems (NIPS)*, volume 25, 2012.

Maximilian Stahlke, Sebastian Kram, Felix Ott, Tobias Feigl, and Christopher Mutschler. Estimating ToA Reliability with Variational Autoencoders. In *IEEE Sensors Journal*, volume 22(6), pp. 5133–5140, March 2022. doi: 10.1109/JSEN.2021.3101933.

Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition. In *Proc. of the ACM Intl. Conf. on Embedded Networked Sensor Systems (SenSys)*, pp. 127–140, November 2015. doi: 10.1145/2809695.2809718.

Petar Stojanov, Zijian Li, Mingming Gong, Ruichu Cai, Jaime G. Carbonell, and Kun Zhang. Domain Adaptation with Invariant Representation Learning: What Transformations to Learn? In *Advances of Neural Information Processing Systems (NIPS)*, 2021.

Baochen Sun and Kate Saenko. Subspace Distribution Alignment for Unsupervised Domain Adaptation. In *Proc. of the British Machine Vision Conf. (BMVC)*, volume 24, pp. 1–10, September 2015. doi: 10.5244/C.29.24.

Baochen Sun and Kate Saenko. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Europ. Conf. on Computer Vision (ECCV)*, volume 9915, pp. 443–450, November 2016. doi: 10.1007/978-3-319-49409-8_35.

Baochen Sun, Jiashin Feng, and Kate Saenko. Correlation Alignment for Unsupervised Domain Adaptation. In *arXiv preprint arXiv:1612.01939v1 [cs.CV]*, December 2016.

Jigang Sun. Curvilinear Component Analysis and Bregman Divergences, April 2010. URL <https://slideplayer.com/slide/12675428/>.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, pp. 4278–4284, February 2017. doi: 10.5555/3298023.3298188.

Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint Optimization Framework for Learning with Noisy Labels. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 5552–5560, Salt Lake City, UT, 2018. doi: 10.1109/CVPR.2018.00582.

Romain Tavenard, Titouan Vayer, François Fleuret, Laetitia Chapel, Nicolas Courty, Rémi Flamary, and Yann Soullard. MAD: Match-And-Deform for Time Series Domain Adaptation. In *Anonymous Submission*, June 2022.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse Sinkhorn Attention. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, pp. 9438–9447, July 2020. doi: 10.5555/3524938.3525813.

Da Teng, Daoerji Fan, Fengshan Bai, and Yuecai Pan. End-to-End Model Based on Bidirectional LSTM and CTC for Online Handwriting Mongolian Word Recognition. In *IEEE Intl. Conf. on Information Science and Technology (ICIST)*, Kaifeng, China, October 2022. doi: 10.1109/ICIST55546.2022.9926844.

James Thornton and Marco Cuturi. Rethinking Initialization of the Sinkhorn Algorithm. In *Proc. of the Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 206, pp. 8682–8698, Valencia, Spain, April 2023.

Naoya Toyozumi, Junji Takahashi, and Guillaume Lopez. Trajectory Reconstruction Algorithm based on Sensor Fusion Between IMU and Strain Gauge for Stand-alone Digital Pen. In *IEEE Intl. Conf. on Robotics and Biomimetics (ROBIO)*, Qingdao, China, December 2016. doi: 10.1109/ROBIO.2016.7866607.

Lloyd N. Trefethen and David Bau. Numerical Linear Algebra. In *SIAM*, April 1997.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep Domain Confusion: Maximizing for Domain Invariance. In *arXiv preprint arXiv:1412.3474v1 [cs.CV]*, December 2014.

Laurens van der Maaten and Geoffrey Hinton. Visualizing Data Using t-SNE. In *Journal for Machine Learning Research (JMLR)*, volume 9(86), pp. 2579–2605, November 2008.

Johannes Rossouw van der Merwe, David Contreras Franco, Jonathan Hansen, Tobias Brieger, Tobias Feigl, Felix Ott, Dorsaf Jdidi, Alexander Rügamer, and Wolfgang Felber. Low-Cost COTS GNSS Interference Monitoring, Detection, and Classification System. In *MDPI Sensors*, volume 23(7), 3452, March 2023. doi: 10.3390/s23073452.

Cédric Villani. Optimal Transport: Old and New. In *Grundlehren der mathematischen Wissenschaften, Springer*, volume 338, September 2008. doi: 10.1007/978-3-540-71050-9.

José R. Villar, Paula Vergara, Manuel Menéndez, Enrique de la Cal, Victor M. González, and Javier Sedano. Generalized Models for the Classification of Abnormal Movements in Daily Life and its Applicability to Epilepsy Convulsion Recognition. In *Intl. Journal Neural Systems*, volume 26(6), pp. 1650037, September 2016. doi: 10.1142/S0129065716500374.

Alessandro Vinciarelli and Michael Peter Perrone. Combining Online and Offline Handwriting Recognition. In *Proc. of the IAPR Intl. Conf. on Document Analysis and Recognition (ICDAR)*, pp. 844–848, Edinburgh, UK, August 2003. doi: 10.1109/ICDAR.2003.1227781.

Qian Wan and Qin Zou. Learning Metric Features for Writer-Independent Signature Verification using Dual Triplet Loss. In *Proc. of the IAPR Intl. Conf. on Pattern Recognition (ICPR)*, pp. 3853–3859, Milan, Italy, January 2021. doi: 10.1109/ICPR48806.2021.9413091.

Jeen-Shing Wang, Yu-Liang Hsu, and Cheng-Ling Chu. Online Handwriting Recognition Using an Accelerometer-Based Pen Device. In *Intl. Conf. on Computer Science and Engineering (CSE)*, July 2013. doi: 10.2991/cse.2013.52.

Junsheng Wang, Tiantian Gong, Zhixiong Zeng, Changchang Sun, and Yan Yan. C³CMR: Cross-Modality Cross-Instance Contrastive Learning for Cross-Media Retrieval. In *Proc. of the ACM Intl. Conf. on Multimedia (ACMMM)*, pp. 4300–4308, Lisbon, Portugal, October 2022. doi: 10.1145/3503161.3548263.

Sinong Wang, Belinda Z. Li, Median Khabsa, Han Fang, and Hao Ma. Linformer: Self-Attention with Linear Complexity. In *arXiv preprint arXiv:2006.04768v3 [cs.LG]*, June 2020.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric Cross Entropy for Robust Learning with Noisy Labels. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, pp. 322–330, Seoul, Korea (South), 2019. doi: 10.1109/ICCV.2019.00041.

Zhiguang Wang, Weizhong Yan, and Tim Oates. Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline. In *arXiv preprint arXiv:1611.06455v4 [cs.LG]*, December 2016.

Lukas Wegmeth, Alexander Hoelzemann, and Kristof Van Laerhoven. Detecting Handwritten Mathematical Terms with Sensor Based Data. In *arXiv preprint arXiv:2109.05594v1 [cs.LG]*, September 2021.

Warren Wiechmann, Robert Edwards, Cheyenne Low, Alisa Wray, Megan Boysen-Osborn, and Shannon Toohey. No Difference in Factual or Conceptual Recall Comprehension for Tablet, Laptop, and Handwritten Note-Taking by Medical Students in the United States: A Survey-based Observational Study. In *Journal Educ. Eval. Health Prof.*, volume 19(8), April 2022. doi: 10.3352/jeehp.2022.19.8.

Robert W. Wiley and Brenda Rapp. The Effects of Handwriting Experience of Literacy Learning. In *Psychol. Sci.*, volume 32(7), pp. 1086–1103, July 2021. doi: 10.1177/0956797621993111.

Garrett Wilson and Diane J. Cook. A Survey of Unsupervised Deep Domain Adaptation. In *Proc. of the ACM Trans. on Intelligent Systems and Technology (TIST)*, volume 11(5), pp. 1–46, October 2020. doi: 10.1145/3400066.

Garrett Wilson, Janardhan Rao Doppa, and Diane J. Cook. Multi-Source Deep Domain Adaptation with Weak Supervision for Time-Series Sensor Data. In *Proc. of the ACM Intl. Conf. on Knowledge Discovery & Data Mining (SIGKDD)*, pp. 1768–1778, August 2020. doi: 10.1145/3394486.3403228.

Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting TF-IDF Term Weights as Making Relevance Decisions. In *ACM Trans. on Information Systems*, volume 26(3), pp. 1–37, June 2008. doi: 10.1145/1361684.1361686.

Yuan Wu, Hongliang Bi, Jing Fan, Guofei Xu, and Huinan Chen. DMHC: Device-free Multi-modal Handwritten Character Recognition System with Acoustic Signal. In *Knowledge-Based Systems*, volume 110314, January 2023. doi: 10.1016/j.knosys.2023.110314.

Meng Xu, Youchen Wang, Bin Xu, Jun Zhang, Jian Ren, Stefan Poslad, and Pengfei Xu. A Critical Analysis of Image-based Camera Pose Estimation Techniques. In *arXiv preprint arXiv:2201.05816v1 [cs.CV]*, January 2022.

Kaihong Yan, Ying Zhang, Haoran Tang, Chengkai Ren, Jian Zhang, Gaoang Wang, and Hongwei Wang. Signature Detection, Restoration, and Verification: A Novel Chinese Document Signature Forgery Detection Benchmark. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5159–5168, New Orleans, LA, June 2022. doi: 10.1109/CVPRW56347.2022.00564.

Xingyi Yang, Xuehai He, Yuxiao Liang, Yue Yang, Shanghang Zhang, and Pengtao Xie. Transfer Learning or Self-supervised Learning? A Tale of Two Pretraining Paradigms. In *arXiv preprint arXiv:2007.04234v1 [cs.CV]*, June 2020.

Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. Towards Accurate Model Selection in Deep Unsupervised Domain Adaptation. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, volume 97, pp. 7124–7133, June 2019.

Mohamed Yousef, Khaled F. Hussain, and Usama S. Mohammed. Accurate, Data-Efficient, Unconstrained Text Recognition with Convolutional Neural Networks. In *Pattern Recognition*, volume 108(107482), December 2020. doi: 10.1016/j.patcog.2020.107482.

Shangshu Yu, Cheng Wang, Yitai Lin, Chenglu Wen, Ming Cheng, and Guosheng Hu. STCLoc: Deep LiDAR Localization With Spatio-Temporal Constraints. In *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*, October 2022. doi: 10.1109/TITS.2022.3213311.

Werner Zellinger, Edwin Lughofer, Susanne Saminger-Platz, Thomas Grubinger, and Thomas Natschläger. Central Moment Discrepancy (CMD) for Domain-Invariant Representation Learning. In *Intl. Conf. on Learning Representations (ICLR)*, April 2017.

George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A Transformer-based Framework for Multivariate Time Series Representation Learning. In *Proc. of the ACM Intl. Conf. on Knowledge Discovery & Data Mining (SIGKDD)*, pp. 2114–2124, August 2021. doi: 10.1145/3447548.3467401.

Hongyu Zhang, Lichang Chen, Yunhao Zhang, Renjie Hu, Chunjuan He, Yaqing Tan, and Jiajin Zhang. A Wearable Real-Time Character Recognition System Based on Edge Computing-Enabled Deep Learning for Air-Writing. In *Journal of Sensors*, volume 2022, May 2022. doi: 10.1155/2022/8507706.

Wenju Zhang, Xiang Zhang, Long Lan, and Zhigang Luo. Maximum Mean and Covariance Discrepancy for Unsupervised Domain Adaptation. In *Neural Processing Letters*, volume 51, pp. 347–366, February 2020a. doi: 10.1007/s11063-019-10090-0.

Xiaoyi Zhang, Jiapeng Wang, Lianwen Jin, Yujin Ren, and Yang Xue. CMT-Co: Contrastive Learning with Character Movement Task for Handwritten Text Recognition. In *Proc. of the IEEE/CVF Asian Conf. on Computer Vision (ACCV)*, volume 13847, pp. 626–642, March 2023. doi: 10.1007/978-3-031-26293-7_37.

Xuchao Zhang, Yifeng Gao, Jessica Lin, and Chang-Tien Lu. TapNet: Multivariate Time Series Classification with Attentional Prototypical Network. In *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, volume 34(4), pp. 6845–6852, New York, NY, April 2020b. doi: 10.1609/aaai.v34i04.6165.

Youshan Zhang. A Survey of Unsupervised Domain Adaptation for Visual Recognition. In *arXiv preprint arXiv:2112.06745v1 [cs.CV]*, December 2021.

Zhilu Zhang and Mart R. Sabuncu. Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. In *Advances of Neural Information Processing Systems (NIPS)*, pp. 8778–8788, Montréal, Canada, 2018. doi: 10.5555/3327546.3327555.

Peilin Zhao, Steven C. H. Hoi, Jialei Wang, and Bin Li. Online Transfer Learning. In *Research Collection School of Information Systems*, volume 216, pp. 76–102, 2014.

Guo-Bing Zhou, Jianxin Wu, Chen-Lin Zhang, and Zhi-Hua Zhou. Minimal Gated Unit for Recurrent Neural Networks. In *Intl. Journal of Automation and Computing (IJAC)*, volume 13, pp. 226–234, June 2016. doi: doi.org/10.1007/s11633-016-1006-2.

Lei Zhu, Jiayu Song, Xiaofeng Zhu, Chengyuan Zhang, Shichao Zhang, and Xinpan Yuan. Adversarial Learning-based Semantic Correlation Representation for Cross-Modal Retrieval. In *IEEE Multimedia*, volume 27(4), pp. 79–90, August 2020a. doi: 10.1109/MMUL.2020.3015764.

Yongchun Zhu, Fuzhen Zhuang, Jindong Wang, Guolin Ke, Jingwu Chen, Jiang Bian, Hui Xiong, and Qing He. Deep Subdomain Adaptation Network for Image Classification. In *IEEE Trans. on Neural Networks and Learning Systems (TNNLS)*, volume 32(4), pp. 1713–1722, May 2020b. doi: 10.1109/TNNLS.2020.2988928.

Bingbing Zhuang and Manmohan Chandraker. Fusing the Old with the New: Learning Relative Camera Pose with Geometry-guided Uncertainty. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 32–42, Nashville, TN, June 2021. doi: 10.1109/CVPR46437.2021.00010.

Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. Supervised Representation Learning with Double Encoding-Layer Autoencoder for Transfer Learning. In *Proc. of the ACM Trans. on Intelligent Systems and Technology (TIST)*, volume 9(2), pp. 1–17, March 2018. doi: 10.1145/3108257.

Xiaowu Zou, Zidong Wang, Qi Li, and Weiguo Sheng. Integration of Residual Network and Convolutional Neural Network Along with Various Activation Functions and Global Pooling for Time Series Classification. In *Neurocomputing*, volume 367, pp. 39–45, November 2019. doi: 10.1016/j.neucom.2019.08.023.

Eidesstattliche Versicherung (Affidavit)

(Siehe Promotionsordnung vom 12. Juli 2011, § 8 Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

München, den 17.09.2023

Ort, Datum

Felix Ott

Glossary

Absolute Pose Regression (APR) is a field of methods for estimating the six degrees of freedom (6DoF) of an object's pose (i.e., position and orientation) in 3D space using machine and deep learning techniques. It is a specific case of pose estimation, which is the process of determining the pose of an object in the real-world from an image input or sequences of images (Kendall et al., 2015). xvii, xix, 22, 23, 25, 26, 283, 299

Character Error Rate (CER) is a common metric to measure the performance of a method that performs character recognition, such as handwriting recognition, optical character recognition, speech recognition, or natural language processing. The CER indicates the percentage of characters that are incorrectly recognized w.r.t. the total number of characters in the test dataset (Kang et al., 2022). xix, 20, 35, 223

Connectionist Temporal Classification (CTC) is a type of neural networks output and is associated with a scoring function utilized for sequence-to-sequence (seq2seq) problems. Examples are classifying words or sentences in the field of handwriting recognition, and speech or audio recognition. A CTC loss function predicts a sequence of labels (including blank outputs) from a sequence of inputs. The CTC loss does not require an explicit alignment between the input and output sequences (Graves et al., 2006). xiii, xix, 19, 21, 169, 204

Convolutional Neural Network (CNN) A CNN is a category of deep learning neural networks that is designed to process data in a grid-like structure. CNNs are often utilized for image inputs. A CNN is composed of pooling layers that reduce the data dimensionality, convolutional layers that perform convolution operations to the input data, and normalization layers that guarantee that the data remains within a range that allows for efficient processing (Goodfellow et al., 2016). xv, xvi, xix, 21, 23, 27, 29–31, 52, 69, 78–81, 83, 86, 87, 89–92, 94, 95, 97, 99, 101, 104, 105, 111, 168, 169, 284

Cross-Modal Retrieval (CMR) is a type of information retrieval in which the data inputs are in different modalities – such as image, text, audio, video, time-series, 3D models, etc. – while retaining a semantically similar context. The primary objective of CMR is to learn a common representation by matching the data modality inputs (Deldari et al., 2022a). xix, 4–8, 21, 32, 33, 47–51, 86, 223

Deep Metric Learning (DML) is a method as a subcategory of representation learning, where the distance or similarity between the representations of different data points can be used to make predictions. The goal is to learn a mapping from the high-dimensional input data to a compact feature space where the distance between data points reflects their semantic similarity (Bengio et al., 2014). v, xix, 4, 6–8, 46–48, 68

Domain Adaptation (DA) is advantageous in real-world scenarios where the data distribution changes over time or in different settings, i.e., the data is not *independent and identically distributed* (i.i.d.). DA is a technique to adapt a model trained on one domain (or distribution of data) to function effectively on a related but different domain. The goal of DA is to improve the model’s performance on the target domain by utilizing knowledge from the source domain (where labeled data is available) to regularize the model for the target domain (where labeled data is scarce) (Wilson & Cook, 2020). xiv–xvii, xix, xxiv, 3–6, 8, 16, 18, 21, 34, 35, 37–42, 44, 45, 47, 48, 51–53, 60, 67–75, 78–82, 86–92, 94, 96–104, 202–204, 266

Domain Shift refers to the dissimilarity in the distribution of input-output pairs between the training and test data. Domain shift occurs when a model (trained on a specific data distribution) is evaluated on a distinct data distribution. This can lead to a decrease in model performance since the model may not generalize well to the new data and requires domain adaptation methods to address this challenge (Wilson & Cook, 2020). v, xvii, 3, 5, 6, 8, 16, 34, 37, 38, 42, 47, 48, 54, 64, 67, 68, 72, 74, 80, 81, 102

Higher-order Moment Matching (HoMM) is a technique to match the higher-order moments of a target distribution to a source distribution up to a certain order p . This is done by minimizing the distance between both distributions (Chen et al., 2020). xvi, xx, 4, 6, 42, 60–62, 70–72, 90, 91, 203

Inertial Measurement Unit (IMU) An IMU is an electronic device that typically includes accelerometers, gyroscopes, and magnetometers. The goal is to measure motions and dynamics of an object, such as its linear acceleration, angular rate, and magnetic field strength (do Monte Lima et al., 2019). xx, 22–24, 26, 32, 33, 111, 134, 169, 298

Long Short-Term Memory (LSTM) is a type of a recurrent neural network that is designed to process entire sequences of data to handle the problem of long-term dependencies in sequential data and vanishing and exploding gradients of the cell. LSTMs use so-called *memory cells* that can maintain its state over a long period of time. This cell selectively chooses which information to keep and which information to discard (Hochreiter & Schmidhuber, 1997). xxi, 24, 25, 28–31, 34, 111, 169

Maximum Mean Discrepancy (MMD) is a non-parametric measure and is defined by representing distances between distributions as distances between mean embeddings of features (first order). MMD is computed by applying a kernel function to the samples of the two distributions (Borgwardt et al., 2006). xiv, xv, xxi, 4, 6, 42, 44, 45, 51–54, 56, 60, 61, 69–72, 79, 88, 90, 91, 95, 99, 102, 104, 203

Modality A modality refers to the type of information or data that is being used in a system or task, e.g., visual data (such as images or videos), audio data, inertial data, GPS data, 3D model data, and text and speech data in natural language processing (Jozé et al., 2020). v, 4, 7, 22, 25, 32–37, 47

Multivariate Time-Series (MTS) An MTS consists of one or multiple variables (i.e., channels) recorded over time. Each variable in an MTS can be thought of as a separate dimension with the goal to analyze and understand the relationships between the different variables over time (Abanda et al., 2022). xxi, xxiv, 5, 6, 8, 18, 20, 29, 31–33, 37, 47, 48, 62, 68, 203

Online Handwriting (OnHW) Recognition is a technology that allows for the recognition of handwriting. This type of recognition is also known as *dynamic* or *real-time* handwriting recognition, and tracks the movement of the stylus or finger on the writing surface to use the resulting data to recognize the written characters, words, or equations. Novel systems use sensor-enhanced pens to gather motion data. OnHW recognition is used for applications such as digital ink input for tablets and smartphones, signature verification, and handwriting-based security systems (Ott et al., 2020b, 2022d). vi, 28, 37, 67

Pairwise Learning is a method where the model is trained to predict the similarity or dissimilarity between pairs of data points. The goal is to learn a mapping from the high-dimensional input data to a lower-dimensional feature space, in which the distance between data points reflects their semantic similarity (Schroff et al., 2015). v, 8

Representation Learning aims to learn a dense, compact, and informative representation of the input data that captures the underlying structure of the data. The objective of representation learning is to find a mapping from the high-dimensional input data to a lower-dimensional space that preserves the most important (and crucial) information and patterns in the data. This learned representation can be utilized as a feature for downstream tasks such as clustering, classification, regression, forecasting, and generation (Bengio et al., 2014). v, vi, 4, 8, 16, 27, 35, 45–47, 62, 68, 95, 204

Self-Localization is the process of predicting the location (position and orientation) of a mobile device, robot, or human within an (unknown) environment. Visual self-localization involves using visual sensors to identify the object's location w.r.t. a

known reference. This information can then be used for various tasks such as mapping, navigation, and localization of other objects or devices (Radwan et al., 2018). v, vi, xvii, 5, 6, 8, 9, 21, 22, 24, 33, 282

Word Error Rate (WER) is a common metric to measure the performance of a method that performs word or sentence recognition, such as handwriting recognition, optical character recognition, speech recognition, text-to-speech synthesis, or natural language processing. The WER indicates the percentage of words that are incorrectly recognized w.r.t. the total number of words in the test dataset (Kang et al., 2022). xxii, 20