Repräsentations- und Transferlernen für Anwendungen des maschinellen Lernens

Robert Kurt Georg Müller

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

Vorgelegt von: Robert Kurt Georg Müller

Tag der Einreichung: 09. Januar 2023

Repräsentations- und Transferlernen für Anwendungen des maschinellen Lernens

Robert Kurt Georg Müller

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

vorgelegt von
Robert Kurt Georg Müller

1. BerichterstatterIn:	Prof. Dr. Claudia Linnhoff-Popien
2. BerichterstatterIn:	Prof. Dr. Alexander Schill
Tag der Einreichung:	09. Januar 2023
Tag der Disputation:	16. Juni 2023

Eidesstattliche Versicherung

(siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Robert Kurt Georg Müller
München, 20.06.2023

Danksagung

Zunächst möchte ich Frau Prof. Dr. Linnhoff-Popien, Inhaberin des Lehrstuhls für Mobile und Verteilte Systeme (LMU München), ganz besonders für die Betreuung meiner Promotion danken. Ihrem Weitblick, Fleiß und Innovationsgeist ist es zu verdanken, dass der Lehrstuhl über eine einmalige Kombination aus Industrie- und Forschungsprojekten verfügt, im Rahmen derer auch diese Dissertation entstand. Ihre Herzlichkeit und ihr Vertrauen ermöglichen ein angenehmes und kollegiales Arbeitsumfeld.
Des Weiteren danke ich Prof. Dr. Schill, Inhaber der Professur für Rechnernetze (TU Dresden), in seiner Rolle als Zweitberichterstatter. Seine freundliche Art und sein schnelles Feedback gaben mir Sicherheit und Zuversicht.
Auch dem Kommissionsvorsitzenden Prof. Dr. Butz und dem Ersatzprüfer Prof. Dr. Schmidt gilt mein Dank.
Ebenfalls vielen Dank meinen Kollegen und Freunden, mit denen ich das Privileg hatte zusammenzuarbeiten. Gerne erinnere ich mich an gemeinsame Forschungsrunden in der Teeküche und an gemütliches Zusammensitzen nach Dienstschluss. Ganz besonderen Dank für die Unterstützung in Forschung, Lehre, Projekten, organisatorischem und privatem möchte ich Steffen Illium, Fabian Ritz, Kyrill Schmid, Michael Kölle, Melisa Delic und Katja Grenner aussprechen.
Zuletzt danke ich meiner Mutter Hella, meinem Vater Jürgen, meiner Schwester Katharina und Bernhard für den bedingungslosen Beistand, den sie mir während der gesamten Zeit haben zukommen lassen.
Grandios - Dankeschön!

Zusammenfassung

Das maschinelle Lernen befasst sich mit dem Lernen von Modellen anhand von Daten. Die Kombination mit neuronalen Netzen wird gemeinhin als Deep Learning bezeichnet und hat zu einem Paradigmenwechsel in fast allen Bereichen der Wissenschaften geführt. Deep Learning wird heutzutage unter anderem zur medizinischen Diagnostik, zur Vorhersage der Proteinfaltung, zur Gesichtserkennung oder sogar zur Schaffung neuer Kunstwerke eingesetzt. Die angesprochenen Anwendungsszenarien sowie ein Großteil der in der Praxis relevanten Datenquellen wie Töne, Videos oder Bilder, sind jedoch hochdimensional. Die direkte Weitergabe an linear Modelle führt, aufgrund des Fluchs der Dimensionalität, in der Regel zu schlechten Ergebnissen. Infolgedessen wurde lange auf das Feature Engineering zurückgegriffen. Anhand von Domänenwissen wird hierbei manuell eine geeignete Menge von Merkmalen extrahiert. Dieser Prozess ist langwierig und kostspielig. Im Gegensatz dazu können neuronale Netze hochdimensionale Daten direkt verarbeiten. Merkmale werden über mehrere Netzschichten hinweg automatisch extrahiert und durch deren Kombination immer spezifischer. Die Aktivierungen einer Schicht können dann als Repräsentation der Eingabe aufgefasst werden. Der Frage, wie ein Netz trainiert werden muss, um gute Repräsentationen extrahieren zu können, widmet sich das *Repräsentationslernen*. Das *Transferlernen* baut darauf auf und beschäftigt sich mit dem Transfer der gelernten Repräsentationen auf nachgelagerte Trainingsaufgaben. Dadurch kann das Wissen vortrainierter Netze effektiv ausgenutzt werden. Die vorliegende Arbeit beschäftigt sich mit dem Repräsentations- und Transferlernen für Anwendungen des maschinellen Lernens. Besonderes Augenmerk liegt dabei auf der Verarbeitung akustischer Signale. Dazu werden zunächst neue Algorithmen und Netzarchitekturen zur Klassifikation von Vokalisationen von Primaten sowie der akustischen Anomalieerkennung vorgestellt, welche die Genauigkeit bisheriger Architekturen übertreffen. Anschließend wird die Eignung des Transferlernens zur akustischen Anomalieerkennung genauer untersucht. Dabei wird gezeigt, dass das Transferlernen die Leistung der Anomalieerkennung steigern kann und dass sich vortrainierte Netze aus unterschiedlichsten Domänen, wie z.B. Musik oder Bildverarbeitung, dazu eignen. Schließlich werden neue Ansätze des Repräsentationslernens für weitere Anwendungsszenarien behandelt. Diese umfassen die diskrete Kommunikation in Multiagentensystemen durch das Clustering der internen Repräsentationen der Agenten sowie das Lernen von Repräsentationen von Fußballteams. In beiden Fällen kann gezeigt werden, dass die vorgestellten Algorithmen vergleichbaren Ansätzen überlegen sind.

Abstract

Machine learning deals with learning models based on sample data. Its combination with neural networks is commonly referred to as Deep Learning and has led to a paradigm shift in almost all areas of science. Deep Learning is being utilized for a variety of tasks, including face recognition, protein folding prediction, medical diagnostics, and even the creation of original art. In general, it should be noted that the mentioned application scenarios as well as a sizable portion of the practical data sources, such as audio, video, or photos, are high dimensional. Directly forwarding the data linear models usually leads to poor results due to the curse of dimensionality. Feature engineering has long been used for effective processing. It involves manually extracting a suitable set of features based on domain knowledge. This process is time-consuming and costly. In contrast, neural networks can process high dimensional data directly. Features are automatically extracted across multiple network layers and become more specific as they are subsequently combined. The activations of a layer can be understood as a representation of the input. The question of how a network must be trained to be able to extract good representations automatically is addressed by the field of *representation learning*. *Transfer learning* expands on this by addressing the transfer of learned representations to downstream tasks, i.e. how pre-trained networks' knowledge can be exploited. This thesis is concerned with representation and transfer learning for machine learning applications. Special attention is given to the processing of acoustic signals. To this end, we first present new algorithms and network architectures for primate vocalization classification and acoustic anomaly detection that outperform the accuracy of previous architectures. Then, the suitability of transfer learning for acoustic anomaly detection is examined in more detail. It is shown that transfer learning can increase the performance of anomaly detection and that pre-trained networks from a wide variety of domains, such as music or image processing, are suitable for this purpose. Finally, we address new approaches to representation learning for further application scenarios. These include discrete communication in multi-agent systems by clustering the agents' internal representations, and learning representations of soccer teams. In both cases, it can be shown that the presented algorithms are superior to other comparable approaches.

Inhaltsverzeichnis

1 Einleitung

Das maschinelle Lernen widmet sich Methoden, die es erlauben, Modelle auf Basis von Beispieldaten zu trainieren. Mit dem gelernten Modell kann anschließend von alten auf neue Fakten geschlossen werden. Im Gegensatz zur manuellen Programmierung, müssen die Entscheidungsregeln so nicht händisch definiert werden, sondern können automatisch anhand der Beispieldaten gelernt werden. Dieser Ansatz ist vor allem dann vorteilhaft, wenn der zu modellierende Sachverhalt komplex ist und eine Vielzahl von Sonderfällen enthält. Es ist z.B. einfacher, ein Modell der Präferenzen eines Kunden aus seinem Einkaufsverhalten abzuleiten, als ihn zu jeder seiner Kaufentscheidungen zu befragen.
Die Kombination des maschinellen Lernens mit künstlichen neuronalen Netzen wird gemeinhin als Deep Learning bezeichnet und ist davon inspiriert, wie Informationen im menschlichen Gehirn verarbeitet werden. Neuronale Netze stellen die aktuell flexibelste und leistungsfähigste Modellklasse des maschinellen Lernens dar.
Der Durchbruch des Deep Learnings ist vor allem der immer weiter steigenden Rechenleistung zu verdanken und hat in fast allen Bereichen der Wissenschaft zu einem Paradigmenwechsel geführt. So ist es z.B. vor kurzem gelungen, die Proteinfaltung, eine der größten Herausforderungen der Wissenschaften, weitestgehend zu lösen [Jum+21]. Überdies wurde Deep Learning eingesetzt, um ein System zu entwickeln [Sil+17], welches die Leistung professioneller Go-Spieler bei weitem übertrifft. Go ist weitaus komplexer als Schach und galt lange Zeit als zu schwierig, um mit maschinellen Lernverfahren gelöst werden zu können.
Bevor das Training neuronaler Netze praktikabel wurde, ging dem Training und der Evaluation von Modellen des maschinellen Lernens das manuelle Feature Engineering voraus. Beim Feature Engineering wird versucht, für die vorliegenden Datenpunkte möglichst hochwertige Repräsentationen (Merkmalsvektoren) zu extrahieren. Weit verbreitete Merkmale in der Bildverarbeitung sind z.B. Farb- und Kantenhistogramme. In der akustischen Signalverarbeitung wird häufig die durchschnittliche Lautstärke, das Tempo und die Tonhöhe verwendet.
Das Feature Engineering ist nötig, da klassische Modelle wie lineare Regressoren oder Support-Vector-Machines unter dem Fluch der Dimensionalität [Bel54; AHK01] leiden. Dieser besagt, dass die Anzahl der Beispiele, die erforderlich sind, um eine beliebige Funktion mit einem bestimmten Genauigkeitsgrad zu schätzen, exponentiell mit der Dimensionalität der Funktion wächst.

Viele Datenquellen wie Töne, Bilder oder Videos sind jedoch inhärent hochdimensional. Die manuelle Extraktion von Repräsentationen für ein vorliegendes Problem ist jedoch langwierig und kostspielig. Als Alternative wird deshalb häufig die automatische Dimensionalitätsreduktion, wie z.B. die Hauptkomponentenanalyse [LKA17], verwendet. Anstelle von Domänenwissen sowie Versuch und Irrtum, basiert diese Methode auf dem Prinzip der Varianzmaximierung. Sie kann aber nur lineare Zusammenhänge modellieren und setzt unabhängig von der Art des Datensatzes auf das gleiche Optimierungsziel. Der Einsatz von neuronalen Netzen zur Dimensionalitätsreduktion stellt einen Kompromiss zwischen manuellem und automatisiertem Feature-Engineering dar und arbeitet datengetrieben. Mit neuronalen Netzarchitekturen ist es möglich, nicht-lineare Zusammenhänge zu modellieren. Sie ermöglichen so die Extraktion von leistungsfähigeren Repräsentationen.
In der Regel wird die hochdimensionale Eingabe durch mehrere Netzschichten propagiert. In jeder Schicht berechnet das neuronale Netz intermediäre Merkmale und kombiniert diese in den darauffolgenden Schichten zu spezifischeren Merkmalen. Auf diese Weise können die Aktivierungen einer jeden Schicht als Repräsentation der Eingabe aufgefasst werden. In den meisten Fällen werden jedoch die Aktivierungen der letzten Schicht verwendet, da diese genau jene Merkmale enthalten, die für die Vorhersage der Ausgabe relevant sind. Häufig folgt auf die letzte Schicht ein linearer Klassifikator oder Regressor. Unter dieser Voraussetzung kann der Prozess auch als Berechnung von linear separierbaren Merkmalen interpretiert werden [GBC16]. Ein neuronales Netz versucht, nicht linear separierbare Eingabedaten, durch das Berechnen von Repräsentationen über mehrere Schichten hinweg, linear separierbar zu machen.
Jede maschinelle Lernaufgabe, die als Modellklasse neuronale Netze verwendet, betreibt demzufolge implizit *Repräsentationslernen*. Infolgedessen können Repräsentationen überwacht oder unüberwacht gelernt werden. Im ersten Fall stehen annotierte Daten zur Verfügung, z.B. bei der Klassifikation von Bildern [Dos+21]. Im zweiten Fall sind keine Annotationen vorhanden und folglich muss das Trainingsziel aus den Daten selbst abgeleitet werden. Dies kann z.B. durch die Vorhersage von Ausschnitten erreicht werden, die aus der Eingabe entfernt wurden [He+22].
In beiden Fällen besteht darüber hinaus die Möglichkeit, die Struktur bzw. die zugrundeliegende Geometrie der Repräsentationen festzulegen [Ayt+18], um den gelernten Merkmalsraum kompakter und separierbarer zu gestalten.
Aufbauend auf dem Repräsentationslernen beschäftigt sich das *Transferlernen* mit der Verwendung der gelernten Repräsentationen für nachgelagerte Lernaufgaben [PY09]. Beim Transferlernen wird ein neuronales Netz nicht für jede Problemstellung von Grund auf neu trainiert, sondern dessen Gewichte werden weiter angepasst oder direkt zu Merkmalsextraktion verwendet. Das Wissen eines bereits vortrainierten Netzes wird also ausgenutzt, um die Leistung der Lernaufgabe in der Zieldomäne zu verbessern. Überdies können so Rechenzeit und Entwicklungsaufwand reduziert werden.

Die vorliegende Arbeit beschäftigt sich mit dem Repräsentations- und Transferlernen für Anwendungen maschinellen Lernens. Der Fokus liegt dabei sowohl auf der Entwicklung neuer Verfahren und Algorithmen als auch dem empirischen Vergleich verschiedener Ansätze für praxisrelevante Problemstellungen. Ein beträchtlicher Teil der angewandten Forschung auf dem Gebiet des Repräsentations- und Transferlernens widmet sich dem computerbasierten Sehen. Intelligente Systeme sollten jedoch in Zukunft möglichst viele Funktionen der menschlichen Sinne abbilden. Zum aktuellen Zeitpunkt sind Methoden zur Verarbeitung akustischer Daten noch weniger erforscht. Die Arbeit leistet ihren Beitrag, diese Forschungslücke weiter zu schließen.
Dazu beleuchtet die vorliegende Arbeit zunächst das Repräsentationslernen für akustische Daten näher. Hierfür wird eine neuronale Netzarchitektur zur Klassifikation von Primatenvokalisationen vorgestellt. Sie kann als wichtiger Baustein eines Wildtier-Informationssystems eingesetzt werden.
Ebenfalls von praktischer Relevanz ist die akustische Anomalieerkennung. In der vorliegenden Arbeit werden dazu Methoden und Algorithmen vorgestellt, mit Hilfe derer Lecks in Wasserversorgungsnetzen und Schäden von Fabrikmaschinen erkannt werden können. Überdies wird die Tauglichkeit des Transferlernens im Kontext der akustischen Anomalieerkennung untersucht.
Die Arbeit beschäftigt sich jedoch ebenfalls mit Anwendungsszenarien, die über die Verarbeitung von akustischen Daten hinausgehen. Hierzu werden Ansätze zur Kommunikation und Absicherung von autonom handelnden Agenten diskutiert. Dabei wird ein besonderes Augenmerk auf die Frage gelegt, wie die Agenten dies auf Basis der gelernten Repräsentationen bewerkstelligen können. Schließlich wird ein Ansatz zum Repräsentationslernen von Fußballmannschaften vorgestellt.
Im Rahmen dieser Arbeit soll die Bedeutung des Repräsentations- und Transferlernens für das angewandte maschinelle Lernen, dargelegt durch zahlreiche Experimente und neu entwickelte Algorithmen, unterstrichen werden.

1.1 Zu Grunde liegende Vorarbeiten

Ein Großteil der Inhalte dieser Arbeit wurde bereits im Rahmen internationaler Fachkonferenzen durch den Autor veröffentlicht. Nachfolgend wird der Eigenanteil des Autors der vorliegenden Arbeit an den für diese Arbeit relevanten und bereits publizierten Inhalten im Einzelnen aufgeschlüsselt. Die einzelnen Vorveröffentlichungen werden dabei in Reihenfolge ihrer ersten inhaltlichen Erwähnung aufgelistet. Zusätzlich zu dieser Übersicht wird zu Beginn der Inhaltskapitel 3, 4 und 5 noch einmal auf die bereits veröffentlichten Inhaltsteile eingegangen. Für alle im Folgenden aufgeführten Publikationen gilt, dass Frau Prof. Dr. Claudia Linnhoff-Popien in ihrer Rolle als Doktormutter des Autors stets mit konstruktiver Kritik und wertvollen Gedanken mitgewirkt hat. Dies gilt insbesondere dann, wenn sie als Autorin aufgeführt ist.

i) **A Deep and Recurrent Architecture for Primate Vocalization Classification [MIL21a]** Diese Publikation stellt eine rekurrente neuronale Netzarchitektur zu Klassifikation von Primatenvokalisationen vor. Sie dient als Grundlage von Kapitel 3.1. Idee, Konzept, Implementierung, Evaluation und Ausarbeitung stammen vom Autor. Steffen Illium half bei der Einordnung und der Diskussion der Ergebnisse.

ii) **Acoustic Leak Detection in Water Networks [Mül+21d]** Im Rahmen dieser Publikation wird ein Verfahren zur akustischen Anomalieerkennung in Wasserversorgungsnetzen vorgestellt. Sie wurde im Rahmen des Projekts „Erkennung und Lokalisierung von Leckstellen in Wassernetzen “ in Zusammenarbeit mit den Stadtwerken München erstellt und bildet die Grundlage von Kapitel 3.2. Die grundlegende Idee und das Konzept stammen vom Autor. Steffen Illium übernahm die Implementierung der Autoencoder. Fabian Ritz und Steffen Illium waren über den gesamten Projektzeitraum an der richtigen Ausrichtung der Hardware und der Erstellung des Datensatzes beteiligt. Darüber hinaus trugen beide durch zahlreiche Diskussionsrunden zur Konkretisierung des Ansatzes bei. Tobias Schröder, Christian Platschek und Jörg Ochs standen vonseiten der Stadtwerken München beratend zur Seite.

iii) **Deep Recurrent Interpolation Networks for Anomalous Sound Detection [MIL21b]** Diese Publikation stellt eine rekurrente neuronale Netzarchitektur zur akustischen Anomalieerkennung in Aufnahmen von Industriemaschinen vor. Kapitel 3.3 basiert auf deren Konzepten und Ergebnissen. Idee, Konzept, Implementierung, Evaluation und Ausarbeitung stammen vom Autor. Steffen Illium half bei der Einordnung und der Diskussion der Ergebnisse. Darüber hinaus trug er mit wichtigen Anmerkungen zur Darstellung der Ergebnisse bei.

iv) **Acoustic Anomaly Detection for Machine Sounds based on Image Transfer Learning [Mül+21c]** In dieser Publikation wird die Tauglichkeit von vortrainierten faltenden neuronalen Netzen zur akustischen Anomalieerkennung untersucht. Alle Netze wurden zur Klassifizierung von Bildern vortrainiert. Diese Arbeit ist als Vorstudie zu der unten folgenden Arbeit [Mül+21a] zu verstehen. Die Inhalte von Kapitel 4.1 basieren auf dieser Publikation. Idee, Konzept, Implementierung, Evaluation und Ausarbeitung stammen vom Autor. Steffen Illium und Fabian Ritz halfen bei der Einordnung und der Diskussion der Ergebnisse.

v) **Analysis of Feature Representations for Anomalous Sound Detection [Mül+21a]** Diese Publikation beschäftigt sich mit dem Transfer von Repräsentationen zur akustischen Anomalieerkennung. Dabei werden die Quelldomänen Musik, Bilder und Geräusche untersucht. Sie basiert auf den Ergebnissen der vorangegangenen Publikation [Mül+21c] und führt

diese konsequent weiter. Die Publikation bildet die Grundlage von Kapitel 4.2. Idee, Konzept, Implementierung, Evaluation und Ausarbeitung stammen vom Autor. Steffen Illium, Fabian Ritz und Kyrill Schmid halfen bei der Einordnung und der Diskussion der Ergebnisse.

vi) **Towards Anomaly Detection in Reinforcement Learning [Mül+22]** Diese Publikation widmet sich ebenfalls der Anomalieerkennung, jedoch im Kontext des Reinforcement Learnings. Die Arbeit unterscheidet sich zu den vorangegangenen Arbeiten darin, dass sie keine konkreten Lösungsansätze und Experimente beinhaltet. Der Schwerpunkt liegt auf der Eingrenzung der Problemstellung, da diese in der Literatur aktuell nicht ausreichend präzise formuliert ist. Kapitel 5.2 basiert auf den Inhalten dieser Publikation. Diese Arbeit ist Teil des Arbeitspaketes „AP3-4: Qualitätssicherung verteilter KI-basierter Systeme“ des Projektes „Unterstützung des thematischen Aufbaus des Instituts für Softwaretechnik Kognitiver Systeme“ in Kooperation mit dem Fraunhofer-Institut für Kognitive Systeme (IKS). Idee, Konzept, Formalisierung und Ausarbeitung stammen vom Autor. Thomy Phan trug mit kritischen Kommentaren und Anregungen zur Konkretisierung und Verbesserung der Idee bei. Einige der in der Arbeit vertretenen Thesen basieren auf zahlreichen Diskussionsrunden mit Steffen Illium und Tom Haider.

vii) **Soccer Team Vectors [Mül+20]** In dieser Publikation wird ein Algorithmus zum Erlernen von Repräsentationen von Fußballmannschaften vorgestellt. Kapitel 5.3 basiert auf dieser Publikation. Idee, Konzept, Implementierung, Evaluation und Ausarbeitung stammen vom Autor. Stefan Langer trug mit wichtigen Anmerkungen zur Motivation der Arbeit bei. Fabian Ritz, Christoph Roch und Steffen Illium halfen bei der Einordnung der Ergebnisse.

1.2 Aufbau dieser Arbeit

In Kapitel 2 werden zunächst wichtige Grundlagen für das weitere Verständnis erörtert. Neben dem maschinellen Lernen wird dabei auch auf die Verarbeitung akustischer Signale eingegangen.
Im anschließenden Kapitel 3 wird das Repräsentationslernen für akustische Daten näher beleuchtet. Hier werden neue Ansätze zu Klassifikation von Primatenvokalisationen sowie der akustischen Anomalieerkennung vorgestellt
Kapitel 4 beschäftigt sich ebenfalls mit der akustischen Anomalieerkennung, legt den Fokus jedoch auf die Anwendung des Transferlernens.
Das letzte Inhaltskapitel 5 widmet sich weiteren Anwendungen des Repräsentationslernens wie dem Multi-Agent Reinforcement Learning und dem Mannschaftssport.

In Kapitel 6 erfolgt schließlich eine Zusammenfassung der wichtigsten Erkenntnisse der vorliegenden Arbeit und ein Ausblick auf mögliche Ansatzpunkte für zukünftige Arbeiten.

2 Grundlagen

Dieses Kapitel vermittelt das nötige Hintergrundwissen, welches die Grundlage für die späteren Kapitel bildet und gliedert sich dabei in zwei Abschnitte: Maschinelles Lernen und akustische Signalverarbeitung.
Das maschinelle Lernen ist ein Teilgebiet der künstlichen Intelligenz und behandelt Methoden, die es erlauben, Modelle auf der Basis von Beispieldaten zu trainieren, um Vorhersagen oder Entscheidungen zu treffen, ohne ausdrücklich dafür programmiert worden zu sein. Da das maschinelle Lernen das Fundament aller in den Folgekapiteln vorgestellten Ansätze bildet, widmet sich Abschnitt 2.1 der Vermittlung von dessen Grundlagen.
Ebenfalls von großer Bedeutung für die vorliegende Arbeit sind die Grundlagen der Verarbeitung von akustischen Signalen, welche in Abschnitt 2.2 besprochen werden. In den folgenden zwei Inhaltskapiteln werden Modelle des maschinellen Lernens für akustische Daten vorgestellt. Hier ist vor allem die Transformation des Roh-Signals in dessen Zeit-Frequenz- bzw. Spektraldarstellung hervorzuheben. Diese dient in den genannten Kapiteln als kompakte Beschreibung des zugrundeliegenden Signals und als Eingabe für die vorgestellten Modelle.
Abschnitt 2.3 fasst schließlich die Erkenntnisse aus diesem Kapitel zusammen. Die wichtigsten Fachbegriffe sind *kursiv* gedruckt.

2.1 Maschinelles Lernen

Die in den Folgekapiteln vorgestellten Verfahren und Algorithmen sind ausnahmslos dem maschinellen Lernen zuzuordnen. Um einen Überblick über das Themengebiet zu vermitteln, werden in Abschnitt 2.1.1 zunächst die Grundlagen mit den wichtigsten Begrifflichkeiten sowie aktuelle Anwendungsmöglichkeiten (Abschnitt 2.1.2) behandelt. Die mathematischen Grundlagen für das Modelltraining folgen in Abschnitt 2.1.3. Da die leistungsfähigsten Algorithmen des maschinellen Lernens auf der Modellklasse der künstlichen neuronalen Netzwerke basieren, werden diese in Abschnitt 2.1.4 genauer diskutiert. Anschließend werden zwei der wichtigsten Netzarchitekturen, faltende (Abschnitt 2.1.5) und rekurrente neuronale Netze (Abschnitt 2.1.6), vorgestellt. Abschließend wird in Abschnitt 2.1.7 darauf eingegangen, wie durch Repräsentationslernen die manuelle Merkmalsextraktion umgangen und mittels Transferlernen Wissen von bereits trainierten neuronalen Netzen ausgenutzt werden kann. Beide Paradigmen kommen in der vorliegenden Arbeit vielfach zum Einsatz.

2.1.1 Grundbegriffe

Als Arthur Samuel 1959 den ersten Algorithmus entwickelte, der bei dem Brettspiel Dame annähernd die Leistung eines Menschen erreichte [Sam59] prägte er den Begriff *Machine Learning* (maschinelles Lernen) als das Forschungsgebiet, welches Computern die Fähigkeit verleiht, zu lernen, ohne explizit programmiert zu werden.
Beim maschinellen Lernen wird Computern also beigebracht, ein Modell der Welt zu lernen, indem Beispiele für den zu lernenden Sachverhalt bereitgestellt werden. Mit dem gelernten Modell kann anschließend von alten auf neue Fakten geschlossen werden. Mit heutigen Verfahren des maschinellen Lernens geschieht dies um mehrere Größenordnungen schneller und in vielen Fällen besser, als es der Mensch leisten könnte.
Dem maschinellen Lernen steht der klassische Ansatz der manuellen Programmierung gegenüber. Hier werden händisch Regeln festgelegt, welche den Sachverhalt möglichst gut modellieren. Sind die Regeln bekannt, einfach zugänglich und von wenigen Sonderfällen geprägt, so kann dieser Ansatz schneller und genauer sein. Es wäre beispielsweise zwecklos, ein Modell zu Addition, Multiplikation oder Subtraktion von reellwertigen Zahlen durch das Bereitstellen von Eingabe-Ausgabe-Paaren zu erlernen. Ist die Beziehung zwischen Eingaben und Ausgaben jedoch komplex und enthält sie eine Vielzahl von Sonderfällen, so ist das maschinelle Lernen der manuellen Programmierung überlegen. Häufig ist es einfacher, Beispiele für das Gewünschte zu liefern, als es in Form von Regeln zu spezifizieren. Soll z.B. ein Modell zur Klassifikation von Tierarten gelernt werden, so wäre es für jede einzelne Tierart nötig, Fachexperten zu befragen und eine Sammlung von Entscheidungsregeln zu erstellen. Zwischen Fröschen und Hunden zu unterscheiden erscheint zwar trivial, um aber entscheiden zu können, ob es sich um einen Gold- oder Streifenschakal handelt, ist jedoch deutlich fundierteres Fachwissen vonnöten. In diesem Fall wäre es einfacher, einen Datensatz aus Bildern und ihren jeweiligen *Labeln* (Kennzeichnungen) zu erstellen und das Modell den Zusammenhang zwischen Bild und Label selbst lernen zu lassen. Allerdings ist für die Erstellung der Labels ebenfalls manueller Aufwand erforderlich, indem Fachexperten den Bildern entsprechende Labels zuweisen. Trotzdem ist dieser Ansatz effizienter, da er sich einfacher parallelisieren lässt, nur eine Entscheidung über das richtige Label benötigt und keine komplexen Regeln definiert und gesammelt werden müssen. Noch einfacher ist es, bestehende Bilddatenbanken zu verwenden oder halb-automatisiert geeignete Webseiten (z.B. Wikipedia) zu durchforsten.

In Abbildung 2.1 werden beide Ansätze visuell verglichen. Der generelle Arbeitsablauf des maschinellen Lernens lässt sich darüber hinaus wie folgt zusammenfassen:

Definition 2.1: Paradigma des maschinellen Lernens

i) Eine Reihe von Beispielen beobachten (Trainingsdaten).

ii) Rückschlüsse auf den Prozess ziehen, der diese Daten erzeugt hat.

iii) Das erlangte Wissen (z.B. gelerntes Modell) nutzen, um Vorhersagen für zuvor nicht gesehene Daten zu machen.

Ferner wird zwischen drei verschiedenen Arten des maschinellen Lernens unterschieden:

i) *Supervised Learning* (Überwachstes Lernen)
Supervised Learning beschreibt den Prozess des maschinellen Lernens, in dem Label für die Beispiele verfügbar und gegeben sind. Es wird also versucht, ein Modell zu lernen, welches für zuvor nicht gesehene Daten das korrekte Label vorhersagt. Zum Trainingszeitpunkt können die Labels genutzt werden, um den Fehler der Vorhersagen des Modells zu quantifizieren. Der Fehler wird wiederum zur Verbesserung des Modells genutzt. Die bekanntesten Vertreter des Supervised Learnings sind die *Klassifikation* und die *Regression*. In Ersterem sind die Labels in Form von Kategorien (z.B. Hund und Katze) gegeben, in Zweiterem in Form von reellwertigen Zahlen oder Vektoren (z.B. Einkommen, Körpergröße).

ii) *Unsupervised Learning* (Unüberwachtes Lernen)
Im Gegensatz zum Supervised Learning, sind beim Unsupervised Learning keine Labels gegeben. Ein Großteil der Algorithmen des Unsupervised Learnings sind dem *Clustering* zuzuordnen. Im Clustering werden verborgene Muster und Informationen in den Daten entdeckt und diese anschließend einem Cluster (Gruppe) zugewiesen. Die Label werden also nicht bereitgestellt, sondern automatisch und ohne Annotationsaufwand, gelernt. Welche Datenpunkte zusammen gruppiert werden, hängt stark von der Wahl der Distanz- oder Ähnlichkeitsfunktion ab. Die Ähnlichkeit zwischen hochdimensionalen Daten wie beispielsweise Bildern oder Audio zu definieren, ist jedoch nicht trivial. Die naive Wahl der Ähnlichkeitsfunktion kann dann dazu führen, dass Bilder nach Farben und nicht nach deren Semantik gruppiert werden.
Viele bewährte Algorithmen des maschinellen Lernens funktionieren nur sehr eingeschränkt mit hochdimensionalen Daten (z.B. Audio, Video, Bilder) aufgrund des Phänomens der *Distanzkonzentration*, welches besagt, dass mit zunehmender Dimensionalität der Daten alle paarweisen Distanzen (Unähnlichkeiten) auf den gleichen Wert konvergieren [AHK01]. Ein weiterer Vertreter des Unsupervised Learnings, die *Dimensionality Reduction* (Dimensionalitätsreduktion), versucht dem entgegenzuwirken. Diese zielt darauf ab, eine kompaktere, niedrigdimensionalere Repräsentation

der Daten zu finden, welche anschließend anstelle der tatsächlichen Daten als Eingabe verwendet werden können. Durch die Dimensionality Reduction werden der Speicherplatzbedarf und die Trainingsdauer verringert, ein gewisser Informationsverlust ist jedoch unvermeidlich.

iii) *Reinforcement Learning* (Bestärkendes Lernen)
Das Reinforcement Learning unterscheidet sich konzeptionell stark von den beiden oben genannten Bereichen des maschinellen Lernens. Während im Supervised- und Unsupervised Learning auf Basis gegebener Beispiele (Daten) gelernt wird, lernt hier ein *Agent* (z.B. Roboter, Computerspielfigur) eine *Policy* (Strategie) durch die Interaktion mit seiner Umgebung. In jedem Zeitschritt beobachtet der Agent den aktuellen Zustand der Umgebung und wählt auf Basis dessen seine nächste Aktion. Anschließend erhält der Agent einen *Reward* (Belohnungssignal), welchen er im Laufe des Trainings durch iteratives Anpassen der Policy maximiert. Eine der größten Herausforderungen ist dabei, dass die Belohnung häufig verzögert auftritt, z.B. erst, wenn das Ziel erreicht ist. Dem Agenten wird also nicht gezeigt, welche Aktion in welchem Zustand am besten ist, sondern er muss dies durch Versuch und Irrtum selbst herausfinden.
Das Reinforcement Learning wird immer dann eingesetzt, wenn automatisch komplexe Strategien oder Verhaltensweisen gelernt werden sollen und es einfacher ist, das Verhalten zu bewerten, als es selbst zu programmieren. Beispielsweise ist es einfacher zu messen, wie lange sich eine Drohne in der Luft halten konnte oder wie vielen Hindernissen sie ausgewichen ist, als die Policy manuell zu definieren.

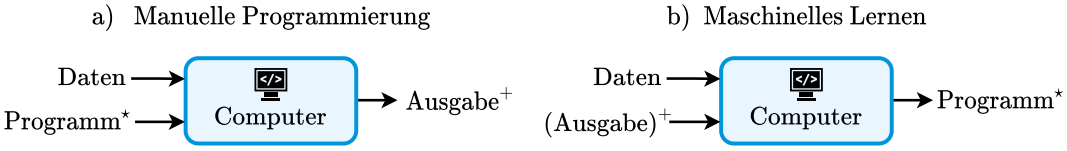


Abbildung 2.1: Visueller Vergleich zwischen der a) manuellen Programmierung und b) maschinellen Lernen. Während bei der manuellen Programmierung das Programm, welches die Daten verarbeitet, um eine Ausgabe zu produzieren, selbst erstellt werden muss, wird im maschinellen Lernen das Programm auf Basis der Eingabedaten und der zugehörigen Ausgabe selbstständig gelernt. Im Fall des Unsupervised Learnings ist die gewünschte Ausgabe (Label) nicht vorgegeben.

2.1.2 Anwendungen

Das maschinelle Lernen ist mittlerweile in fast jeden Bereich des alltäglichen Lebens vorgedrungen und hat in den letzten Jahren, bedingt durch wachsendes

Forschungsinteresse, für immer ausgereiftere und komplexere Anwendungsmöglichkeiten gesorgt.
Es ist nun möglich, realistische Profilbilder von Personen zu generieren die nicht existieren [KLA21], alte Bilder fotorealistisch zu restaurieren [Wan+21] sowie neue Musikstücke [Dha+20] oder ganze Texte [Bro+20] künstlich zu erzeugen.
Schon seit längerem wird maschinelles Lernen eingesetzt, um Spam E-Mails zu erkennen und noch genereller eine E-Mail auf Basis ihres Inhalts in entsprechende Kategorien wie Werbung oder Benachrichtigungen einzusortieren [Cra+15].
Die Bekämpfung der Verbreitung von Hatespeech [MPH22] (Hassrede), insbesondere in den sozialen Medien, wäre ohne den Einsatz von Algorithmen des maschinellen Lernens aufgrund der Menge an anfallenden Daten undenkbar.
Recommender Systems [Sou+21] (Empfehlungssysteme) lernen mit Hilfe von großen Mengen von Nutzerdaten, personalisiert neue Lieder, Filme oder Produkte vorzuschlagen. Dies geschieht indem das Verhalten ähnlicher Nutzer mit einbezogen wird.
Chatbots [CJM22] werden heutzutage eingesetzt, um die Kundenberatung zu vereinfachen und den Personalbedarf zu reduzieren. Darüber hinaus kommt kein Sprachassistentent (z.B. Siri, Google Assistent) auf heutigen Smartphones ohne maschinelles Lernen aus.
AlphaGo [Sil+16] war das erste System welches, basierend auf einer Mischung aus Supervised- und Reinforcement Learning, einen professionellen Go-Spieler schlagen konnte. Aufgrund des größeren Spielfeldes und der höheren Anzahl an möglichen Zügen weist Go eine größere Komplexität als Schach auf. Maschinelle Lernsysteme erzielten bis zur Entwicklung von AlphaGo nur eine mittelmäßige Leistung, die nicht an die eines menschlichen Go-Spielers heranreichte. Das Nachfolgesystem *AlphaGoZero* [Sil+17] beseitigte AlphaGo's Abhängigkeit von zuvor aufgezeichneten Spielzügen. AlphaGoZero verfügt bis auf die Spielregeln über keinerlei externe Informationen und lernt ausschließlich mittels Reinforcement Learning. Indem das System immer wieder gegen sich selbst spielt (*Self-Play*), lernt es immer komplexere Handlungsstrategien. Die Autoren ließen AlphaGo und AlphaGoZero 100 mal gegeneinander antreten und konnten zeigen, dass AlphaGoZero jedes einzelne Spiel gewinnt.
Dass maschinelles Lernen die menschliche Leistung in hochkomplexen Brettspielen übertrifft, ist zwar wichtig, um den Fortschritt der Forschung und die damit einhergehenden neuen Möglichkeiten zu demonstrieren, hat aber nur eingeschränkten gesamtgesellschaftlichen Nutzen. Im Gegensatz dazu wird maschinelles Lernen in Anwendungen der medizinischen Bildverarbeitung verwendet, um die Entscheidungsfindung von Medizinern zu unterstützen und zu verbessern. Es kann eingesetzt werden, um schnell zu überprüfen, ob Röntgenaufnahmen relevante Anomalien enthalten und so die Bearbeitungszeit um bis zu 28% senken [Nab+21]. Da gezeigt werden konnte, dass Fachexperten mit zunehmendem Zeitdruck die Genauigkeit bei der Erkennung von metastasie-

rendem Brustkrebs bei Sentinel-Lymphknoten-Biopsien rapide sinkt [Bej+17], wurde ein Algorithmus entwickelt [Liu+19], welcher diesen mit einer Genauigkeit von 99% erkennen kann. Darüber hinaus wurden Fortschritte im Bereich des maschinellen Lernens genutzt, um SARS-CoV-2-Varianten im Hinblick auf die Immunflucht (Escape-Variante) einzustufen [Beg+23].
Mit *AlphaFold2* [Jum+21] wurde kürzlich ein maschinelles Lernsystem vorgestellt, welches die Proteinfaltung, eine der größten Herausforderungen der Wissenschaften, weitestgehend löst. Proteine sind die Grundlage jedes biologischen Prozesses und ein Protein besteht aus einer Sequenz von Aminosäuren, die über Peptidbindungen miteinander verbunden sind. Die Wechselwirkungen zwischen den Aminosäuren bewirken die Faltung des Proteins und ergeben so seine natürliche dreidimensionale Struktur. Durch die computergestützte Prognose der Faltung muss diese wesentlich seltener experimentell im Labor untersucht werden und verspricht somit eine enorme Beschleunigung des wissenschaftlichen Fortschritts auf dem Gebiet der Biologie und Chemie, z.B. um Krankheiten schneller zu erforschen und Medikamente zu deren Bekämpfung zu entwickeln.
Ein weiterer Anwendungsfall, vor allem im industriellen Kontext, ist die Anomalieerkennung. Hier wird ein Modell, das den Regelbetrieb modelliert, gelernt und dazu verwendet, Abweichungen davon festzustellen. So können zum Beispiel Motorengeräusche, autonome Roboter, der Zahlungs- und Netzwerkverkehr oder der Gesundheitszustand eines Patienten überwacht werden [Pan+21].
Die angesprochenen Anwendungen dienen der Verdeutlichung der Relevanz und Allgegenwart des maschinellen Lernens und stellen nur einen Ausschnitt der mannigfaltigen Anwendungsmöglichkeiten dar.

2.1.3 Maximum-Likelihood-Schätzung von Modellparametern

Anhand der in Kapitel 2.1.2 vorgestellten Anwendungen des maschinellen Lernens lässt sich feststellen, dass die gelernten Modelle entweder generativ oder diskriminativ arbeiten. Während diskriminative Modelle eine Entscheidungsgrenze lernen, modellieren generative Modelle die gesamte Datenverteilung und erlauben es, neue Datenpunkte zu erzeugen.
Seien $\mathbf{x}$ und $\mathbf{y}$ Zufallsvariablen für die Daten bzw. die Labels, dann lernt ein diskriminatives Modell die bedingte Wahrscheinlichkeit $p(\mathbf{y}|\mathbf{x})$ und ein generatives Modell die gemeinsame Verteilung $p(\mathbf{x}, \mathbf{y})$ oder nur $p(\mathbf{x})$, wenn keine Labels gegeben sind. Ein Großteil der in der jüngeren Literatur sowie in der vorliegenden Arbeit vorgestellten maschinellen Lernverfahren basiert auf der Maximum-Likelihood Methode zum Schätzen der unbekannten Modellparameter θ wie z.B. den Gewichten eines neuronalen Netzes oder den Parametern einer Wahrscheinlichkeitsverteilung. Ein Modell ist definiert über eine parametrisierte Wahrscheinlichkeitsverteilung $p(\cdot|\theta)$, wobei die Modellparameter

anhand eines gegebenen Datensatzes geschätzt werden.
Der Datensatz lässt sich als Stichprobe der zugrundeliegenden unbekannten (wahren) Verteilung interpretieren. Ein Datensatz von tausenden von Rocksongs oder Profilbildern kann so als Stichprobe der Verteilung aller Rocksongs bzw. aller Profilbilder aufgefasst werden. Es ist also häufig sehr viel einfacher aus einer Verteilung Stichproben zu ziehen, als auf diese direkt zuzugreifen.
Für den Fall eines generativen Modells ohne Labels sei die unbekannte (wahre) Verteilung $p(\mathbf{x}|\theta^\star)$. Ein Datensatz $\mathcal{D}$ der Länge n stellt eine Stichprobe aus $p(\mathbf{x}|\theta^\star)$ dar:

$$\mathcal{D} = \{x_i | 0 \leq i \leq n\}, \quad \forall x_j \in \mathcal{D} : x_j \sim p(\mathbf{x}|\theta^\star) \tag{2.1}$$

Das Ziel ist es nun, $p(\mathbf{x}|\theta^\star)$ mittels $p(\mathbf{x}|\theta)$ so gut wie möglich anzunähern. Es gilt zu beachten, dass die Parameter $\theta^\star$ fixiert und unbekannt sind und eine möglichst gute Annäherung durch θ gefunden werden soll. Diese Problemstellung lässt sich als Minimierungsproblem der *Kullback-Leibler-Divergenz* [KL51] (KL-Divergenz) zwischen den beiden Verteilungen formulieren:

$$\arg\min_{\theta} D_{\mathrm{KL}}\left(p(\mathbf{x}|\theta^\star) \,||\, p(\mathbf{x}|\theta)\right) \tag{2.2}$$

Die KL-Divergenz beruht auf dem Konzept der *Entropie* einer Verteilung $\mathcal{H}(p(\mathbf{x}))$. Die Entropie ist ein Maß der Unordnung, Zufälligkeit und Unvorhersagbarkeit.
In der Informationstheorie quantifiziert sie den durchschnittlichen Informationsgehalt einer Verteilung. Der Informationsgehalt einer einzelnen Stichprobe $x_i \sim p(\mathbf{x})$ ist definiert durch die einhergehende Reduktion der Unsicherheit, welche durch deren Beobachtung eintritt [Sha48]. Der Faktor, um den sich die Unsicherheit reduziert, ist das Reziproke der Wahrscheinlichkeit der Stichprobe $\frac{1}{p(x_i)}$. Da in der Informationstheorie Information in Bit gemessen wird (1 Bit = Reduktion der Unsicherheit um den Faktor zwei), ist der Informationsgehalt der Stichprobe x_i gegeben durch
$\log_2(\frac{1}{x_i}) = \log_2(1) - \log_2(x_i) = -\log_2(x_i)$.
Im maschinellen Lernen wird jedoch üblicherweise der natürliche Logarithmus (Basis e) verwendet und Information in *Nats* (Natural Unit of Information) gemessen, da der tatsächliche absolute Wert der Information für Optimierungs- und Minimierungsprobleme unerheblich ist. Die Entropie ist folglich gegeben durch:

$$\mathcal{H}(p(\mathbf{x})) = -\mathbb{E}_{p(\mathbf{x})}[\log(p(\mathbf{x}))] \tag{2.3}$$

Abbildung 2.2 dient der Veranschaulichung des Konzepts. Während sich die Entropie auf eine einzelne Wahrscheinlichkeitsverteilung bezieht, kann die *Kreuzentropie* (Cross Entropy) $\mathcal{H}(p(\mathbf{x}), q(\mathbf{x}))$ verwendet werden, um die Abweichung zwischen zwei Wahrscheinlichkeitsverteilungen zu quantifizieren. Sie bezeichnet die durchschnittliche Anzahl von Nats, die benötigt wird, wenn bei der Berechnung von $\mathcal{H}(p(\mathbf{x}))$ zur Bestimmung des Informationsgehalts eine

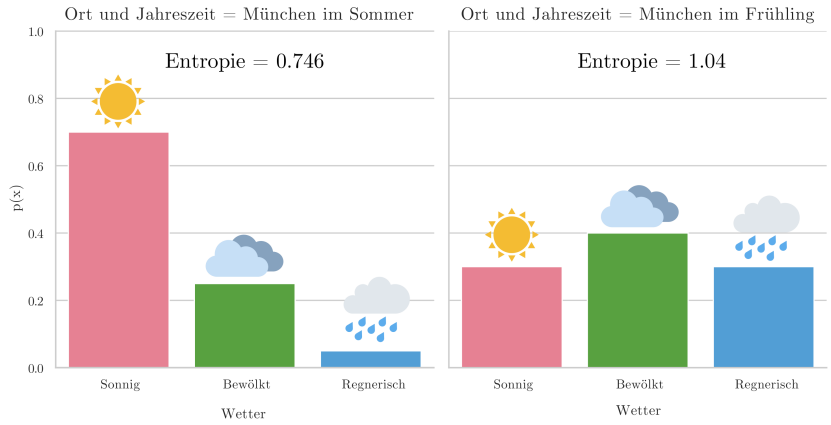


Abbildung 2.2: Beispiel für die Entropie zweier kategorischer Wahrscheinlichkeitsverteilungen. Im Sommer in München ist sonniges Wetter sehr viel wahrscheinlicher als bewölktes oder regnerisches Wetter. Die dazugehörige Wahrscheinlichkeitsverteilung (links) ist entsprechend spitz und unausgeglichen. Die Entropie der Verteilung ist gering, da das Wetter einfach vorherzusagen ist. Im Frühling (rechts) ist die Verteilung gleichmäßiger (fast uniform). Die Entropie ist höher, da das Wetter im Frühling wechselhafter ist und sich schwieriger vorhersagen lässt.

andere Verteilung $q(\mathbf{x})$ (z.B. ein Modell) verwendet wird:

$$\mathcal{H}(p(\mathbf{x}), q(\mathbf{x})) = -\mathbb{E}_{p(\mathbf{x})}[\log(q(\mathbf{x}))] \tag{2.4}$$

Daraus folgt direkt, dass wenn $q(\mathbf{x}) = p(\mathbf{x})$, dann ist

$$\mathcal{H}(p(\mathbf{x}), q(\mathbf{x})) = \mathcal{H}(p(\mathbf{x}), p(\mathbf{x})) = -\mathbb{E}_{p(\mathbf{x})}[\log(p(\mathbf{x}))] = \mathcal{H}(p(\mathbf{x}))$$

und somit gilt $\mathcal{H}(p(\mathbf{x}), q(\mathbf{x})) \geq \mathcal{H}(p(\mathbf{x}))$.
Die Kreuzentropie ist ein absolutes Maß. Um ein relatives Maß zu erhalten genügt es, die Entropie $\mathcal{H}(p(\mathbf{x}))$ zu subtrahieren. Damit wird die durch $q(\mathbf{x})$ verursachte zusätzliche Anzahl an Nats quantifiziert. Dies ist genau die Definition der KL-Divergenz:

$$\begin{aligned} D_{\mathrm{KL}}(p(\mathbf{x}) \,||\, q(\mathbf{x})) &= \mathcal{H}(p(\mathbf{x}), q(\mathbf{x})) - \mathcal{H}(p(\mathbf{x})) \\ &= \mathbb{E}_{p(\mathbf{x})}[\log(p(\mathbf{x})) - \log(q(\mathbf{x}))] \\ &= \mathbb{E}_{p(\mathbf{x})}\left[\log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right)\right] \end{aligned} \tag{2.5}$$

Die KL-Divergenz ergibt null, wenn beide Verteilungen gleich sind und ist umso größer, je ungleicher diese sind. Sie ist jedoch kein Distanzmaß, da sie die Eigenschaft der Symmetrie nicht erfüllt $D_{\mathrm{KL}}(p(\mathbf{x}) \,||\, q(\mathbf{x})) \neq D_{\mathrm{KL}}(q(\mathbf{x}) \,||\, p(\mathbf{x}))$.

Mit den oben genannten Definitionen lässt sich nun das Minimierungsproblem aus Gleichung 2.2 wie folgt vereinfachen:

$$
\begin{aligned}
&\arg\min_{\theta} \; D_{\mathrm{KL}}\left(p(\mathbf{x}|\theta^{\star}) \,||\, p(\mathbf{x}|\theta)\right) \\
&= \arg\min_{\theta} \; \mathbb{E}_{p(\mathbf{x}|\theta^{\star})}\left[\log(p(\mathbf{x}|\theta^{\star})) - \log(p(\mathbf{x}|\theta))\right] \\
&= \arg\min_{\theta} \; \cancel{\mathbb{E}_{p(\mathbf{x}|\theta^{\star})}\left[\log(p(\mathbf{x}|\theta^{\star}))\right]} - \mathbb{E}_{p(\mathbf{x}|\theta^{\star})}\left[\log(p(\mathbf{x}|\theta))\right] \qquad (2.6) \\
&= \arg\min_{\theta} \; -\mathbb{E}_{p(\mathbf{x}|\theta^{\star})}\left[\log(p(\mathbf{x}|\theta))\right] \qquad (2.7)
\end{aligned}
$$

Da der erste Term in Gleichung 2.6 nicht von den Parametern θ abhängt, über die minimiert wird, kann er ignoriert werden. Der Erwartungswert aus Gleichung 2.7 erfordert jedoch vollen Zugriff auf $p(\mathbf{x}|\theta^{\star})$. Wie zu Beginn bereits angesprochen, ist diese Annahme generell unrealistisch und es steht nur eine endliche Menge von Stichproben (Gleichung 2.1) zur Verfügung. Mit Hilfe der Stichproben kann der Erwartungswert approximiert werden. So ergibt sich folgendes Minimierungs- bzw. Optimierungsziel:

$$
\begin{aligned}
&\arg\min_{\theta} \; -\frac{1}{n}\sum_{i=1}^{n} \log(p\left(x_i|\theta\right)) \qquad (2.8) \\
&= \arg\max_{\theta} \; \frac{1}{n}\sum_{i=1}^{n} \log(p\left(x_i|\theta\right)) \qquad (2.9)
\end{aligned}
$$

Es wird also lediglich die logarithmische Wahrscheinlichkeit der Stichproben (Beispiele, Beobachtungen) unter dem Modell maximiert. Dieses Verfahren ist in der Literatur als Maximum-Likelihood-Methode bekannt.

2.1.4 Neuronale Netze

Im vorangegangenen Kapitel wurde die Interpretation eines Modells als parametrisierte Wahscheinlichkeitsverteilung und dessen Optimierung mit Hilfe der Maximim-Likelihood Methode vorgestellt. Dabei wurde die tatsächliche Gestalt des Modells bewusst offen gelassen. Bevor ein konkretes maschinelles Lernproblem in die Praxis umgesetzt werden kann, ist es jedoch unabdingbar, die Modellklasse festzulegen.
Die in Kapitel 2.1.2 diskutierten Fortschritte basieren fast ausschließlich auf der Modellklasse der tiefen neuronalen Netze, mit Hilfe derer beliebige Eingabe-Ausgabe-Funktionen gelernt werden können [HSW89]. Ein Großteil der in dieser Arbeit vorgestellten Ansätze und Methoden basiert ebenfalls auf tiefen neuronalen Netzen. Maschinelles Lernen auf Basis von tiefen neuronalen Netzen wird als *Deep Learning* bezeichnet. Im Folgenden soll auf die grundlegende Funktionsweise neuronaler Netze genauer eingegangen werden.

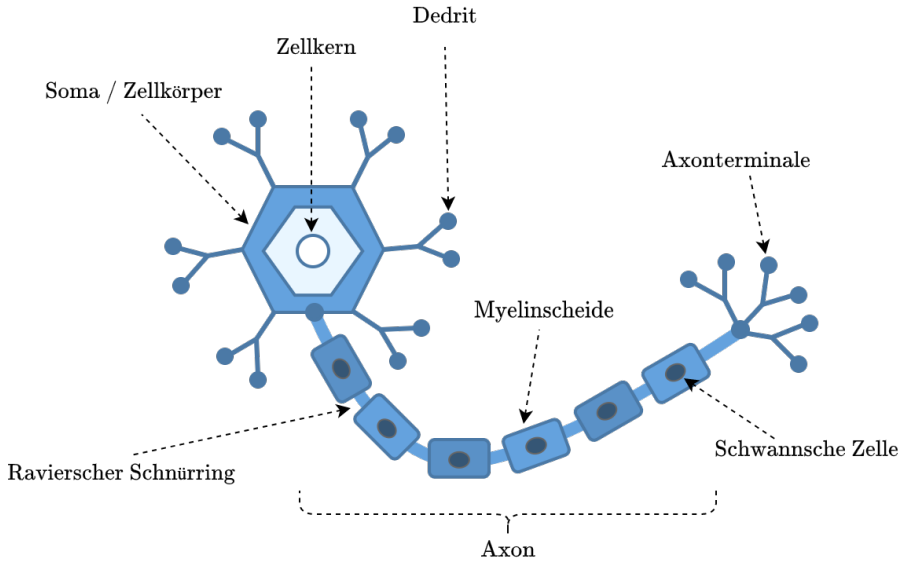


Abbildung 2.3: Schematische Darstellung eines Neurons. Die Struktur folgt [Pur+01].

Obwohl die Forschung auf dem Gebiet bis ins Jahr 1949 zurückreicht [Heb49], ist es erst seit Kurzem aufgrund steigender Rechenleistung möglich neuronale Netze in annehmbarer Zeit zu trainieren. Verantwortlich hierfür ist das Moore-sche Gesetz [Moo+65], das besagt, dass sich die Anzahl der Transistoren in einem integrierten Schaltkreis etwa alle zwei Jahre verdoppelt.
Der erste große Durchbruch gelang Krizhevsky et al. [KSH12] 2012. Die Autoren entwickelten ein neuronales Netz für die ImageNet Large Scale Visual Recognition Challenge zur Klassifikation von Bildern, welches den nächstbesten Konkurrenten bezüglich seiner Genauigkeit um 41 % übertraf. Somit wurde erstmalig gezeigt, dass neuronale Netze eine praktikable Modellklasse sind, die das Potenzial haben, bewährte Ansätze in ihrer Leistungsfähigkeit zu übertreffen. In den darauf folgenden Jahren wurde dadurch die explosionsartige Verbreitung des Deep Learnings ausgelöst.
Die Funktionsweise neuronaler Netze ist davon inspiriert, wie Neuronen im menschlichen Gehirn Informationen verarbeiten. Das menschliche Gehirn umfasst etwa 90 Milliarden Neuronen, wobei jedes Neuron mit tausenden von anderen Neuronen verbunden ist. Ein Neuron besteht aus drei Hauptteilen: Zellkörper, Dendriten und Axon und ist die primäre Funktionseinheit des Nervensystems. Ein Neuron ist in Abbildung 2.3 grafisch dargestellt. Rezeptoren an den Dendriten empfangen chemische Signale von anderen Neuronen. Für die Signalübertragung zwischen Neuronen sind biochemische Botenstoffe (Neurotransmitter) zuständig. Die eingehenden Signale verursachen elektrische Veränderungen im Neuron, die vom Zellkern interpretiert werden. Wenn die Summe der elektrischen Eingänge einen bestimmten Schwellenwert überschreitet, feuert das Neuron und gibt den Impuls über das Axon weiter. Dieses Signal

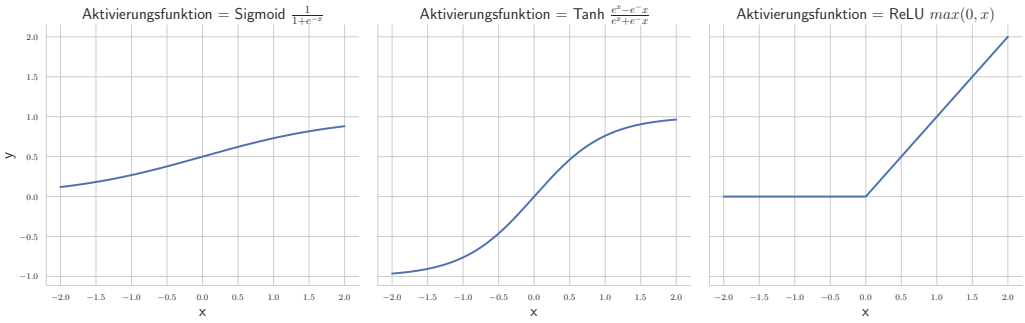


Abbildung 2.4: Drei gängige Aktivierungsfunktionen. Sigmoid (links) und Tangens Hyperbolicus (mitte) stauchen die Werte zwischen null und eins bzw. minus eins und eins, die Rectified Linear Unit (rechts) setzt alle Werte kleiner null auf null.

wird Aktionspotenzial genannt. Das Ende des Axons besteht aus der Axonterminale, die wiederum mit den Dendriten anderer Neuronen verbunden ist. Wenn das Aktionspotenzial das Axonterminale erreicht, wird die Freisetzung des Neurotransmitters ausgelöst, welcher die Dendriten der angeschlossenen Neuronen stimuliert. [Pur+01]
Ein künstliches Neuron [Ros58] bildet die grundlegende Funktionsweise eines Neurons mathematisch ab. Es erhält n reellwertige Eingabesignale, gewichtet und summiert diese und wendet eine *Aktivierungsfunktion* an, um das Ausgabesignal zu berechnen. Das Ausgabesignal kann wiederum als Teil des Eingabesignals eines Folge-Neurons dienen.
Seien $x \in \mathbb{R}^n$ die n Eingabewerte (Signale), $w \in \mathbb{R}^n$ der Gewichtsvektor zur Gewichtung der einzelnen Eingabewerte, $b \in \mathbb{R}$ das Potenzial (*Bias*) und $\sigma : \mathbb{R} \to \mathbb{R}$ eine Aktivierungsfunktion, dann berechnet sich die Ausgabe y eines einzelnen künstlichen Neurons wie folgt:

$$y = \sigma(x \cdot w + b) \tag{2.10}$$

Der Bias[1] legt dabei das bereits vorhandene Aktivierungspotential eines Neurons fest. Mit der Aktivierungsfunktion σ lässt sich die Ausgabe in einen gewünschten Wertebereich (z.B. $y \in [0, 1]$) überführen, um eine Wahrscheinlichkeit zu modellieren. Darüber hinaus ist es durch die Verwendung von geeigneten Aktivierungsfunktionen möglich, nicht-lineare Zusammenhänge abzubilden. Beispiele für einige weit verbreitete Ativierungsfunktionen sind in Abbildung 2.4 zu finden. Die Ausgabe y wird häufig auch *Aktivierung* genannt. Aus der Zusammensetzung künstlicher Neuronen und deren Organisation in Schichten resultiert das *Feedforward neuronale Netzwerk* (FFNN). Durch dessen hierarchische Schichtstruktur und die Verwendung vieler Neuronen lassen

[1] Der Bias wird aus Gründen der Übersichtlichkeit in den folgenden Abbildungen nicht explizit aufgeführt.

sich so komplexere Funktionen abbilden. Die erste und letzte Schicht wird als Eingabe- bzw. Ausgabeschicht bezeichnet, dazwischen befinden sich die versteckten Schichten. Abbildung 2.5 verdeutlicht exemplarisch die Struktur eines FFNNs.
Seien $a_l \in \mathbb{R}^n$ die Aktivierungen (Ausgabe) von Schicht l, $W_{l+1} \in \mathbb{R}^{d \times n}$ die Gewichtsmatrix wobei

$$W_{l+1} = \begin{bmatrix} — & w_{l+1}^{(1)} & — \\ — & \dots & — \\ — & w_{l+1}^{(d)} & — \end{bmatrix} = \begin{bmatrix} w_{l+1}^{1,1} & \cdots & w_{l+1}^{1,n} \\ \vdots & \ddots & \vdots \\ w_{l+1}^{d,1} & \cdots & w_{l+1}^{d,n} \end{bmatrix} \tag{2.11}$$

und $b_{l+1} \in \mathbb{R}^d$ der Bias Vektor. Jeder Zeilenvektor $w_{l+1}^{(i)}$ beschreibt die Gewichte der Verbindungen von Neuron i in Schicht $l+1$ zu den Aktivierungen a_l. So berechnet sich die Ausgabe von Schicht a_{l+1} wie folgt:

$$a_0 = \text{Eingabedaten} \tag{2.12}$$

$$z_{l+1} = W_{l+1} a_l + b_{l+1} \tag{2.13}$$

$$a_{l+1} = \sigma_l(z_{l+1}) \tag{2.14}$$

Die Aktivierungen werden demzufolge Schicht für Schicht durch das neuronale Netzwerk propagiert, bis die Aktivierungen der Ausgabeschicht berechnet sind. Dieser Prozess wird als *Forward Pass* bezeichnet.
Die Parameter eines neuronalen Netzes mit Ein- und Ausgabeschicht sowie L versteckten Schichten sind gegeben durch $\theta = \{W_1, b_1, \dots W_L, b_L, W_{L+1}, b_{L+1}\}$. Um ein neuronales Netz zu trainieren, werden zunächst alle Gewichte in θ zufällig initialisiert. Folglich wird die berechnete Ausgabe zunächst nicht mit der gewünschten übereinstimmen. Um die Parameter so anzupassen, dass das Netz die Ausgaben richtig vorhersagt, muss der Fehler mittels einer skalaren Fehlerfunktion E quantifiziert werden.
Eine passende Fehlerfunktion kann über die Maximum-Likelihood-Methode aus Kapitel 2.1.3 hergeleitet werden. Der Gradient der Fehlerfunktion in Bezug auf die Parameter gibt die Richtung des steilsten Anstiegs der *Fehlerfunktion* (Loss Function) an. Um die Fehlerfunktion zu minimieren, wird das Verfahren des Gradientenabstiegs verwendet. Das Gradientenabstiegsverfahren bewegt sich iterativ mit Lernrate α im Parameterraum in die entgegengesetzte Richtung des Gradienten.

$$\theta' \leftarrow \theta - \alpha * \frac{\partial E}{\partial \theta} \tag{2.15}$$

Dabei ist die Lernrate α ist ein wichtiger Hyperparameter, da zu hohe Werte dazu führen können, dass das Minimum verfehlt wird, während eine zu niedrige Lernrate das Training unnötig verlangsamt. Zur Verbesserung der Konvergenz hat sich die Verwendung einer adaptiven Lernrate durch Algorithmen wie AdamW [LH18] durchgesetzt.

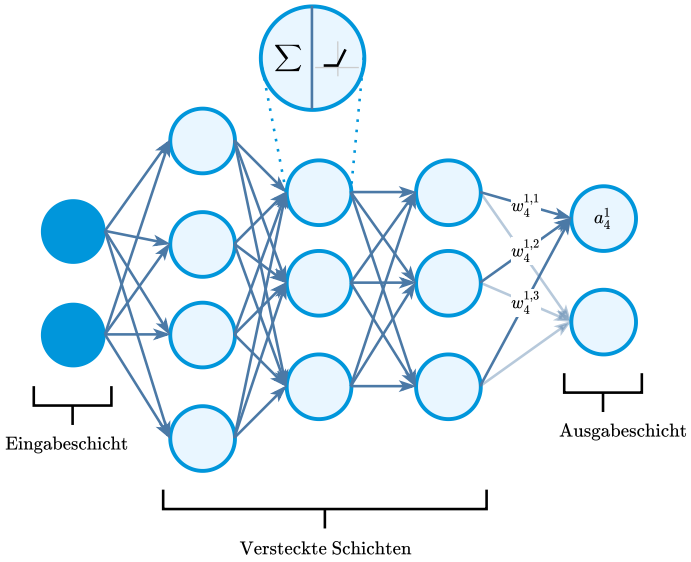


Abbildung 2.5: Ein Feedforward neuronales Netzwerk (FFNN) mit Eingabe- und Ausgabeschichten der Größe zwei sowie drei versteckten Schichten.

Für die effiziente Berechnung des Gradienten werden heutzutage automatische Differenzierungsverfahren [Pas+19] basierend auf dem *Backpropagation* Algorithmus [RHW86] verwendet. Backpropagation kann den Gradientenabstieg um einen Faktor von bis zu zehn Millionen beschleunigen [Web15; GW08].
Die automatische Differenzierung ist ein Verfahren, das Ableitungen mit Maschinenpräzision berechnet, ohne explizit einen symbolischen Ausdruck für die gesamte Ableitung zu bilden. Es beruht auf der Anwendung der Kettenregel, um komplexe Funktionen in elementare Funktionen zu zerlegen, für die die Ableitung exakt berechnet werden kann. Die automatische Differenzierung eignet sich deshalb besonders, da ein neuronales Netz als Komposition von einfachen Funktionen aufgefasst werden kann, siehe Gleichung 2.14.
Dazu muss zunächst der gerichtete und azyklische Berechnungsgraph des Forward Passes aufgestellt werden. Um den Gradienten der Fehlerfunktion E zu berechnen, muss für jeden Knoten v_i im Berechnungsgraph die partielle Ableitung von v_i in Bezug auf E ($\frac{\partial E}{\partial v_i}$) berechnet werden. Diese lässt sich mit Hilfe der Kettenregel der multivariaten Analysis auf dem zugrundeliegenden Berechnungsgraphen wie folgt rekursiv berechnen:

$$\frac{\partial E}{\partial v_i} = \sum_{j \in \mathtt{K}(v_i)} \frac{\partial E}{\partial v_j} \frac{\partial v_j}{\partial v_i} \tag{2.16}$$

wobei $\mathtt{K}(v_i)$ die Menge der Kindnoten von v_i beschreibt. Durch den Start der Prozedur am Ende des Berechnungsgraphens liegen die partiellen Ableitungen des Fehlers in Bezug auf die Kindknoten bereits vor und müssen folglich nicht neu berechnet werden.

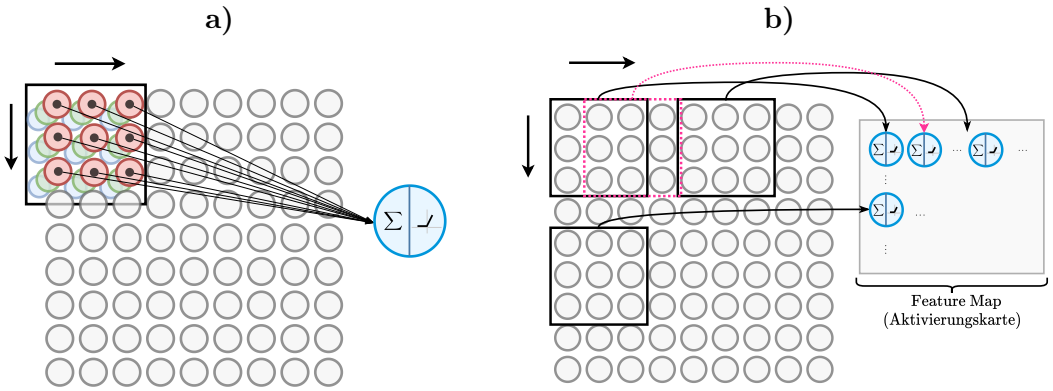


Abbildung 2.6: a) Ein 3×3 Filter berechnet die Aktivierung für den linken oberen Ausschnitt eines RGB-Bildes. b) Der 3×3 Filter bewegt sich mit Schrittweite 1 über das Bild und berechnet die Feature Map. Die räumliche Struktur des Bildes bleibt erhalten.

2.1.5 Faltende neuronale Netze

Faltende neuronale Netze [LeC+99] (Convolutional Neural Networks, kurz *CNNs*) kombinieren Faltungsmatrizen, die in der digitalen Bildverarbeitung für Filter (z.B. Gauß- und Kantenfilter) verwendet werden, mit neuronalen Netzen und sind besonders für den Einsatz mit Bilddaten geeignet.
Ein klassisches neuronales Netz, wie in Kapitel 2.1.4 vorgestellt, verbindet jedes Pixel eines Bildes mit den Neuronen der ersten versteckten Schicht. Diese Struktur stößt schnell an ihre Grenzen, da die Dimension der Eingabeschicht mit steigender Bildgröße quadratisch anwächst (Höhe $\times$ Breite $\times$ Farbraum). Gleichzeitig wächst auch die Anzahl an benötigten Parametern des neuronalen Netzes stark an. Ein RGB-Bild der Größe $256 \times 256 \times 3$ benötigt demzufolge schon 196.608 Dimensionen in der Eingabeschicht. Darüber hinaus trägt dieser Ansatz den speziellen Strukturen eines Bildes nicht Rechnung. Indem ein Bild als Vektor aus Pixelwerten aufgefasst wird, bricht die räumliche Struktur des Bildes auf. Obwohl ein FFNN durchaus in der Lage ist diese in den versteckten Schichten wieder zu rekonstruieren [Tol+21], ist es effizienter, die Bildstruktur in der Netzarchitektur direkt zu berücksichtigen.
Der Aufbau eines CNNs ähnelt dem Vernetzungsmuster des visuellen Cortex. Einzelne Neuronen reagieren nur auf Reize in einem begrenzten Bereich des gesamten Sichtfeldes, dem sogenannten rezeptiven Feld. Eine Vielzahl derartiger Felder überlappen sich und decken so den gesamten visuellen Bereich ab [Bij+18]. Es gilt also das Prinzip der Lokalität, ein einzelnes Neuron im visuellen Cortex hat nur Zugriff auf einen kleinen Teil aller eingehenden Informationen. Dieser Sachverhalt wird in einem CNN durch die Verwendung von trainierbaren Filtern umgesetzt. Im ersten Schritt muss die Höhe und Breite des Filters festgelegt werden. Üblicherweise werden kleine quadratische Filter (3×3, 5×5 oder 7×7) verwendet. Der Filter wird dann mit einer horizontalen und einer vertikalen Schrittweite (*Stride*) von links oben nach rechts unten

über das Bild beweg, vgl. Abbildung 2.6b). In jedem Schritt multipliziert der Filter seine Gewichte mit den darunterliegenden Pixeln (inneres Produkt). Dabei gilt es zu beachten, dass der Filter sich über die gesamte Tiefe des erstreckt. Allgemeiner gesprochen, erstreckt sich der Filter über die gesamte Tiefe des Eingabevolumens. Die Geometrie eines Filters in einem CNN ähnelt folglich der eines Würfels. Ein 3×3 Filter für ein RGB-Bild (drei Farbkanäle) hat demnach $3 * 3 * 3 = 27$ trainierbare Gewichte. Nach der Multiplikation werden alle Werte summiert und eine Aktivierungsfunktion angewendet. Ein Filter verbindet somit in einem kleinen lokalen Fenster jeweils ein einzelnes Neuron mit jedem Pixel des Fensters. Folglich ist das Ergebnis einer einzelnen Anwendung der skalare Wert der Aktivierung des Neurons, vgl. Abbildung 2.6a). Für ein Fenster der Tiefe T, Höhe B und Breite B sowie der Aktiverungsfunktion σ werden für den darunterliegenden Bildausschnitt x die Gewichte des Filters w und b wie folgt angewandt:

$$\sigma(\sum_{i=1}^{T}\sum_{j=1}^{H}\sum_{k=1}^{B} x_{i,j,k} * w_{i,j,k} + b_{i,j,k}) \tag{2.17}$$

Die Aktivierungen, berechnet durch das Bewegen des Filters über das gesamte Bild, werden wiederum in der gleichen zweidimensionalen Struktur angeordnet. Die resultierende Matrix von Aktiverungswerten wird *Feature Map* (Aktivierungskarte) genannt. Abbildung 2.6b) stellt die Berechnung einer Feature Map visuell dar. Die Größe (Höhe und Breite) der resultierenden Feature Map entspricht in etwa der des Ausgangsbildes. Je nach Filter- und Schrittgröße kann es vorkommen, dass an den Bildkanten nicht mehr genügend Pixel zur Verfügung stehen, um den Filter zu füllen. Im einfachsten Fall werden diese Stellen ignoriert, wodurch sich die Größe der Feature Map leicht reduziert. Eine andere Möglichkeit ist das Auffüllen mit einem konstanten Wert (*Padding*), meist mit Nullen, womit die Größe der Feature Map die gleiche Größe wie das zugrundeliegende Bild aufweist.
Die resultierenden Aktivierungen messen den Grad des Vorhandenseins des Filters, z.B. einer vertikalen Kante. Der Filter repräsentiert ein gewisses Merkmal und die Aktivierung beschreibt die Intensität des Merkmals. In der Praxis werden in einer Schicht mehrere Filter angewendet, um eine größere Menge an Feature Maps zu berechnen, welche entlang der Tiefendimension übereinander gelegt werden. In der darauffolgenden Schicht wird dieselbe Prozedur mit neuen Filtern auf dem Volumen (gestapelte Feature Maps) der letzten Schicht wiederholt. So lernt das Netzwerk durch die Schichten immer komplexere Zusammenhänge zu erkennen. In den ersten Schichten eines CNNs bilden sich zunächst einfache Kanten- und Farbfilter. Die Aktivierungen dieser Filter werden in den mittleren Schichten wiederum verwendet und kombiniert, um daraus spezifischere Muster wie Waben- oder Kreisformen zu erkennen. Generell ist zu beobachten, dass der Grad der Spezialisierung mit der Tiefe des CNNs zunimmt und es einem CNN ermöglicht, immer komplexere Strukturen

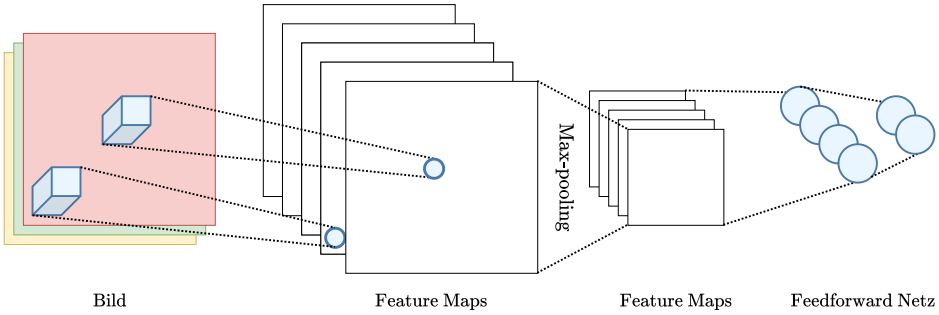


Abbildung 2.7: Beispiel für die Bildverarbeitung mit einem CNN. Jeder (trainierbare) Filter produziert eine neue Feature Map für die nächste Schicht. Mittels Max-Pooling wird die Größe der Feature Maps reduziert. Schlussendlich verarbeitet ein FFNN die kleineren Feature Maps und berechnet die Ausgabe.

und Objekte in einem Bild zu erkennen. So können in den letzten Schichten zum Beispiel Filter gelernt werden, die Hundeköpfe erkennen [OMS17].
Ein Problem der Aktivierungen der Feature Maps besteht darin, dass sie die genaue Position der Merkmale im Eingabebild erfassen. Dies führt dazu, dass kleine Veränderungen in der Position des Merkmals im Bild zu einer veränderten Feature Map führen. Ein Lösungsansatz zur Verringerung der Empfindlichkeit ist das Downsampling der Feature Map, um so lokale Translationsinvarianz herzustellen. Dies bewirkt, dass die resultierenden Feature Maps robuster gegenüber Änderungen der Position des Merkmals im Bild sind. Dieses Verfahren wird als *Pooling* bezeichnet. Beim Max-Pooling wird immer nur die höchste Aktivierung innerhalb eines kleinen Pooling-Fensters übernommen, beim Average-Pooling wird das arithmetische Mittel der Aktivierungen berechnet. Das Pooling Fenster wird wie beim Filter über die gesamte Feature Map geschoben. In den meisten Fällen wird das Max-Pooling der Größe 2×2 mit Schrittweite 2 verwendet, was die Größe der Feature Map um den Faktor 2 reduziert. Darüber hinaus lässt sich durch die Verwendung von Pooling die Anzahl der benötigten Gewichte des CNNs reduzieren. Durch mehrmaliges Pooling zwischen den Schichten des CNNs lässt sich schlussendlich wieder ein FFNN anschließen, um das Bild beispielsweise zu klassifizieren.

2.1.6 Rekurrente neuronale Netze

Datenquellen mit sequentieller oder temporaler Struktur sind allgegenwärtig. DNA, Protein- und Gensequenzen, EKG-Messungen, Text, Audio, Video, Bewegungstrajektorien oder Zeitreihen wie Aktienkurse sind nur einige Beispiele für deren weite Verbreitung.
Eine Sequenz ist eine Auflistung fortlaufend nummerierter Datenpunkte (Zeitschritte). In praktischen Anwendungsszenarien hängt die Ausprägung eines Datenpunktes von den vorangegangenen Datenpunkten ab. Diese Eigenschaft

wird als *autoregressiv* bezeichnet.
Mit einem FFNN ist die Verarbeitung einer Sequenz nur durch die Eingabe der Konkatenation der Datenpunkte möglich. Dieses Vorgehen führt zum einen dazu, dass die Anzahl der zu berücksichtigenden Zeitschritte festgelegt werden muss und nicht variabel ist, und zum anderen dass die Anzahl der benötigten Gewichte abhängig von der Anzahl der Zeitschritte ist. So wird die Modellgröße schnell unpraktikabel.
Rekurrente neuronale Netze (RNNs) erweitern das FFNN, indem sie Rückkopplungen zwischen einzelnen Schichten erlauben. Sie sind besonders für die Verarbeitung von Sequenzen geeignet.
RNNs liegt die Idee zugrunde, für jeden Zeitschritt der Sequenz ein FFNN mit den gleichen Gewichten zu verwenden (*Parameter Sharing*) und zusätzlich einen versteckten Zustand zu führen, welcher als differenzierbarer Speicher verstanden werden kann. Dieser kann zu jedem Zeitschritt aktualisiert und dem FFNN für den nächsten Zeitschritt übergeben werden. Dadurch wird der Informationsfluss von einem Zeitschritt zum Nächsten ermöglicht. So lassen sich Sequenzen variabler Länge bei gleichbleibender Anzahl von Gewichten verarbeiten. Weiterhin wird so die sequentielle Struktur der Eingabedaten direkt in der Netzarchitektur berücksichtigt.
In einem RNN ist der versteckte Zustand h_{t+1} eine Funktion f über der Eingabe des aktuellen Zeitschritts x_t und dem vorangegangenen versteckten Zustand h_{t-1} und definiert somit die folgende Rekurrenzrelation:

$$h_t = f(x_t, h_{t-1}) \tag{2.18}$$

Das RNN unterhält also immer einen versteckten Zustand h_t, der die vergangene Sequenz von Eingaben zusammenfasst. Dies ist in Abbildung 2.8 auch visuell abgebildet. Der initiale versteckte Zustand h_0 wird dabei auf den Nullvektor $\mathbf{0}_d$ gesetzt.
Sei $h_t \in \mathbb{R}^{d_{out}}, x_t \in \mathbb{R}^{d_{in}}$ der versteckte Zustand respektive die Eingabe, dann ist ein klassisches RNN wie folgt parametrisiert:

$$h_t = \tanh(Wh_{t-1} + b_h + Ux_t + b_x). \tag{2.19}$$

wobei $W, U \in \mathbb{R}^{d_{out} \times d_{in}}$ und $b_h, b_x \in \mathbb{R}^{d_{out}}$. Um eine Ausgabe zu berechnen ist es üblich, h_t als Eingabe für ein FFNN zu nutzen.
Die Sequenzmodellierung mit RNNs erweitert das Spektrum der Einsatzmöglichkeiten um folgende Szenarien, die sequentielle Eingaben und gegebenenfalls auch sequentielle Ausgaben beinhalten können:

- Many to one (m:1)
  Hier erhält das RNN eine Sequenz von Datenpunkten als Eingabe und produziert eine einzelne Ausgabe. Der häufigste Anwendungsfall ist die Klassifikation von Sequenzen.

- One to many (1:m)

Hier wird für eine einzelne Eingabe eine Sequenz von Ausgaben erzeugt. Dies wird vor allem für generative Aufgaben verwendet, wie zum Beispiel der Erstellung von Bildbeschreibungen oder der Text- und Musiksynthese.

- Many to many (m:n)
  Hier erhält das RNN eine Sequenz von Datenpunkten als Eingabe und produziert eine Ausgabesequenz. Die Länge der Ausgabesequenz muss dabei nicht zwangsläufig der Länge der Eingabesequenz entsprechen. Typische Beispiele sind die maschinelle Übersetzung und die Eigennamenerkennung.

Backpropagation in einem RNN wird auf das in der Zeit ausgerollte Netz angewandt und *Backpropagation Through Time* genannt. Der rechte Teil von Abbildung 2.8 verdeutlicht dies. Prinzipiell sind rekurrente neuronale Netze in der Lage, langfristige Abhängigkeiten zwischen weit entfernten Zeitschritten zu modellieren. Es wurde jedoch gezeigt [Hoc91], dass RNNs in der Praxis aufgrund des Problems des verschwindenden und explodierenden Gradienten in dieser Hinsicht scheitern. Beim verschwindenden Gradienten ist die Auswirkung von versteckten Zuständen aus früheren Zeitschritten verschwindend gering. Beim explodierenden Gradienten führen große Gradienten zu instabilem Training.
Bei der genaueren Betrachtung der Berechnung des Gradienten aus Gleichung 2.16 wird ersichtlich, dass der Gradient der ersten Zeitschritte vom Produkt der Gradienten der späteren Zeitschritte abhängt und der multiplikative Faktor mit steigender Sequenzlänge für explodierende oder verwschwindende Gradienten sorgt.
Um das Problem zu entschärfen, entwickelten Hochreiter und Schmidhuber 1997 das *Long Short-Term Memory* [HS97] (LSTM). Das LSTM verringert das Problem der Nichtberücksichtigung von Langzeitabhängigkeiten. Das LSTM kontrolliert den Informationsfluss explizit durch die Verwendung differenzierbarer Gatter (Gates) und fördert das Behalten von Informationen aus früheren Zeitschritten durch die Verwendung eines zusätzlichen internen Zustands. Dies geschieht, indem die Berechnungen eines einfachen RNNs durch eine Reihe komplexerer Operationen ersetzt werden.

$$i_t = \text{sigmoid}(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \tag{2.20}$$

$$f_t = \text{sigmoid}(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \tag{2.21}$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \tag{2.22}$$

$$o_t = \text{sigmoid}(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \tag{2.23}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{2.24}$$

$$h_t = o_t \odot \tanh(c_t) \tag{2.25}$$

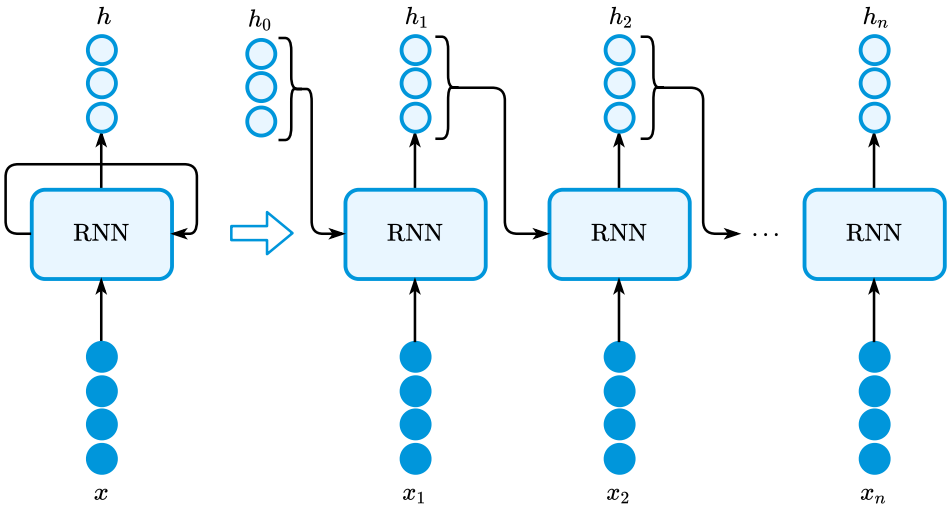


Abbildung 2.8: Ein rekurrentes neuronales Netz, das entlang der Zeitachse ausgerollt ist.

Zusätzlich zu einem verborgenen Zustand h_t verfügt ein LSTM auch über einen Zellzustand c_t. Der Zellzustand wird als differenzierbarer Speicher modelliert, der relevante Informationen über die gesamte Sequenz hinweg behält. c_t agiert als Langzeitgedächtnis (Long-Term Memory) und h_t als Kurzzeitgedächtnis (Short-Term Memory). Das Lesen, Schreiben und Löschen von Informationen des Zellzustands wird durch die folgenden Gatter geregelt:

- *Input Gate* (Eingabegatter, Gleichungen 2.20 und 2.22)
  Gleichung 2.20 erhält als Eingabe die Konkatenation von h_{t-1} und x_t und modelliert eine weiche[2] Entscheidung darüber, zu welchem Grad die einzelnen Werte des Zellzustands zu aktualisieren sind. Gleichung 2.22 erzeugt eine Reihe von Kandidatenwerten für den neuen Zellstatus.

- *Forget gate* (Vergessengatter, Gleichung 2.21)
  Das Forget Gate erhält als Eingabe die Konkatenation von x_t und h_{t-1} und modelliert eine weiche Entscheidung darüber, welche Werte des Zellzustands zu vergessen sind.

- *Cell Gate* (Zellgatter, Gleichung 2.24)
  Der neue Zellzustand c_t berechnet sich durch die Addition von zwei Teilen. Der erste Teil, das Hadamard Produkt (elementweise Multiplikation) des Forget Gates f_t und des aktuellen Zellzustands c_{t-1}, modelliert das Löschen von Werten aus c_{t-1}. Der zweite Teil, das Hadamard Produkt des Input Gates i_t und den Kandidatenwerten g_t für den neuen Zellzustand, beschreibt die neu in den Zellzustand zu schreibenden Werte. c_t ist folglich eine Kombination aus alten Werten des Zellzustands die erhalten bleiben, und neu berechneten Werten.

[2] Kontinuierlich, ein Wert $\in [0, 1]$ pro Element in c_{t-1}

- *Output Gate* (Ausgabegatter, Gleichungen 2.23 und 2.25)
  Auf Basis der Eingabe x_t und dem vorangegangenen versteckten Zustand h_{t-1} modelliert Gleichung 2.23 eine weiche Entscheidung darüber, zu welchem Grad die Werte des neuen Zellzustands c_t in den versteckten Zustand h_t einfließen. h_t wird in Gleichung 2.25 berechnet und enthält nur einen Teil des gesamten (aktivierten) Zellzustands (z.B. nur für den nächsten Zeitschritt wichtige Informationen).

Durch die additive Berücksichtigung neuer Informationen, vgl. Gleichung 2.24, wird der multiplikative Faktor, der die Gradienten verschwinden oder explodieren lässt, abgeschwächt.
Seit der Entwicklung der LSTMs wurden viele Verbesserungen und Erweiterungen entwickelt. Die zwei wichtigsten Vertreter sind hier die *Gated-Recurrent-Unit* (GRU) [Cho+14] sowie der Attention Mechanismus [BCB15]. Die GRU vereinfacht die Berechnung des LSTMs, indem sie Gatter zusammenlegt. Attention berechnet zusätzlich eine Score für die vorangegangenen versteckten Zustände. Anschließend wird anhand der Score eine gewichtete Summe der vorangegangenen Zustände erstellt. Darüber hinaus kann ein LSTM auch bidirektional [GJM13] oder mit mehreren übereinander gelegten Schichten operieren.

2.1.7 Repräsentations- und Transferlernen

Vor der Verwendung von Algorithmen des maschinellen Lernens steht immer die Frage, welche Daten als Eingabe dienen und wie diese repräsentiert werden sollen. Die Leistung der Algorithmen hängt fundamental von der richtigen Beantwortung dieser Frage ab. Es wird hierbei zunächst zwischen *Rohdaten* und *Merkmalsvektoren* unterschieden.
Rohdaten wurden noch nicht für die Weiterverarbeitung aufbereitet und liegen in der Form vor, in der sie anfallen (z.B. unkomprimierte Audio- oder Videodaten).
Merkmalsvektoren sind n-dimensionale Vektoren, die eine kompakte numerische Beschreibung der zugrundeliegenden Rohdaten darstellen, z.B. in einem Tonsignal durch die durchschnittliche, minimale und maximale Energie oder durch die Farb- und Kantenhistogramme von Bildern. Es gilt jedoch zu beachten, dass diese Unterscheidung nicht immer möglich ist. Tabulare Daten wie die Leistungsparameter eines Sportlers liegen häufig schon in Rohform als Merkmalsvektoren vor. Es ist jedoch möglich, auf Basis der Merkmalsvektoren weitere Merkmale zu berechnen, z.B. mit Polynomfunktionen.

Bevor der Einsatz von neuronalen Netzen aufgrund von mangelnder Rechenleistung und fehlender Forschung auf dem Gebiet praktikabel wurde, ging dem Training und der Evaluation das manuelle *Feature Engineering* voraus. Beim Feature Engineering wird versucht, für die vorliegenden Rohdaten auf Basis von Expertenwissen möglichst hochwertige Merkmalsvektoren, häufig auch

Repräsentationen genannt, zu extrahieren. Dies ist nötig, da klassische Modelle wie lineare Regressoren, Support-Vector-Machines oder Gaussian-Mixture-Models unter dem *Curse of Dimensionality* (Fluch der Dimensionalität) [Bel54] leiden. Dieser besagt, dass die Anzahl der Beispiele, die erforderlich sind, um eine beliebige Funktion mit einem bestimmten Genauigkeitsgrad zu schätzen, exponentiell mit der Dimensionalität der Funktion wächst. Zudem nimmt die durchschnittliche Norm eines Vektors mit unabhängigen und identisch verteilten Dimensionen logarithmisch mit der Anzahl der Dimensionen zu, während die Varianz konstant bleibt. Das heißt, die Abstände in hochdimensionalen Räumen werden immer größer und ähnlicher und damit weniger aussagekräftig. Viele der wissenschaftlich interessanten Datenquellen wie Ton, Bilder oder Videos sind jedoch inhärent hochdimensional. Da die manuelle Extraktion von geeigneten Repräsentationen langwierig und teuer ist, wird als Alternative häufig die automatische Dimensionalitätsreduktion verwendet. Lange Zeit wurde hierfür die Hauptkomponentenanalyse angewendet. Auch dieses Verfahren projiziert die Datenpunkte in einen niedriger dimensionierten Unterraum. Anstelle von Domänenwissen sowie Versuch und Irrtum, basiert diese Methode auf dem Prinzip der Varianzmaximierung. Die Hauptkomponentenanalyse kann jedoch nur lineare Zusammenhänge modellieren und setzt unabhängig von der Art der Daten auf das gleiche Optimierungsziel. Sie ist folglich nicht datengetrieben.
Der Einsatz von neuronalen Netzen zur Dimensionalitätsreduktion stellt einen Kompromiss zwischen manuellem und automatisiertem Feature-Engineering dar. Außerdem ist es mit modernen Netzarchitekturen möglich, nicht-lineare Zusammenhänge zu modellieren und die Dimensionalitätsreduktion mit variierenden Optimierungszielen durchzuführen.
Möglichst generelle und aussagekräftige Repräsentationen automatisiert für Rohdaten zu extrahieren, ist die Aufgabe des *Repräsentationslernens* (Representation Learning).
Die folgenden Eigenschaften charakterisieren die Güte gelernter Repräsentationen:

- Entflechtet (*Disentangled*)
  Jede Dimension der Repräsentation ist genau einem Merkmal der Daten zugeordnet. Die einzelnen Dimensionen sollen also möglichst statistisch unabhängig voneinander sein.

- Kompakt und kohärent
  Die extrahierten Repräsentationen sind möglichst niedrigdimensional. Sich ähnelnde Datenpunkte sollen auch in dem gelernten Merkmalsraum benachbart sein.

- Transferierbar
  Die Repräsentationen sollen nicht nur für eine einzige Aufgabe, sondern für möglichst viele verschiedene Aufgaben einsetzbar sein.

Durch die hierarchische Berechnung und Extraktion von Merkmalen der Eingabedaten können die Aktivierungen jeder Schicht eines trainierten neuronalen

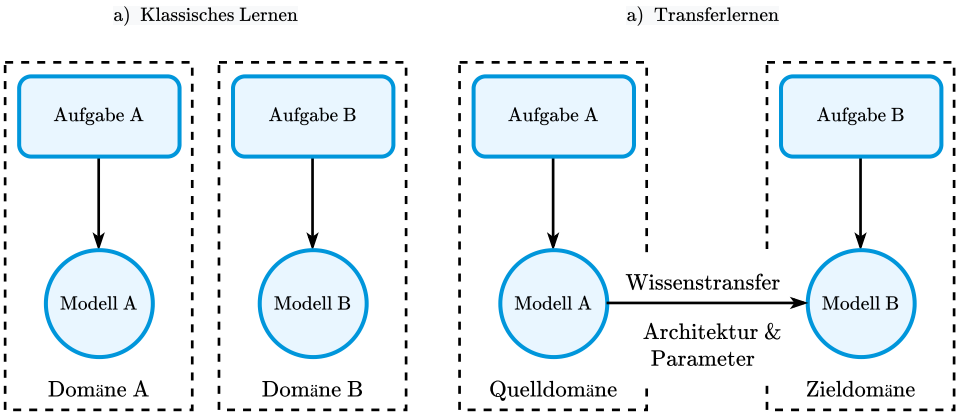


Abbildung 2.9: Vergleich des klassischen Ansatzes zum Lernen von Modellen und dem Transferlernen. Während beim klassischen Lernen Aufgaben isoliert betrachtet werden, nutzt das Transferlernen Wissen der Quelldomäne aus, um eine Lernaufgabe in der Zieldomäne zu verbessern.

Netzes als Repräsentationen aufgefasst werden. Jede maschinelle Lernaufgabe, die als Modellklasse neuronale Netze verwendet, betreibt demzufolge implizit Repräsentationslernen. Die Güte der Repräsentationen ist vor allem im Bereich des Supervised Learnings, häufig nicht der Forschungsschwerpunkt. Solange ein Klassifikator eine hohe Genauigkeit für den untersuchten Datensatz erzielt, sind die genannten Charakteristika oftmals vernachlässigbar.
Im Gegensatz dazu, beschäftigt sich das *Self-Supervised Learning* [Car+19; Che+20; Zbo+21; He+22] (Selbstüberwachtes Lernen, SSL) explizit mit dem Lernen von Repräsentationen in Abwesenheit von Labels. SSL generiert Labels aus den Daten selbst, wobei häufig die zugrunde liegende Struktur der Daten genutzt wird, z.B. durch die Vorhersage maskierter Segmente [He+22].

Während im Repräsentationslernen untersucht wird, wie geeignete Repräsentationen gelernt werden können, beschäftigt sich das *Transferlernen* (Transfer Learning) mit der Frage, welche Repräsentationen für welche nachgelagerten Aufgaben geeignet sind und wie sich diese für weitere Aufgaben effizient anpassen lassen [PY09]. Das Weiterverwenden von bereits trainierten Modellen und deren erlerntem Wissen kann in vielen Fällen die enormen Rechen- und Zeitressourcen reduzieren, die für die Entwicklung und das Training von Modellen benötigt werden (Abbildung 2.9). Dabei wird zwischen zwei Paradigmen unterschieden:

- *Finetuning*
  Die Parameter des Netzes werden durch das Training mit neuen Datenpunkten angepasst. Es besteht die Gefahr, dass zuvor erlangtes Wissen überschrieben wird (*Catastrophic Forgetting*).

- *Feature-Extraction* (Merkmalsextraktion)
  Das Netz wird nicht weiter trainiert, es werden lediglich die Aktivierungen einer oder mehrerer zuvor festgelegter Schichten extrahiert. Bei dieser Methode kann dem Netz kein neues Wissen beigebracht werden.

Beim Transferlernen wird Wissen (Netzarchitektur und Parameter), welches durch das Training einer Ausgangsaufgabe in einer Quelldomäne gesammelt wurde, auf eine Zieldomäne für eine Zielaufgabe transferiert. Die Quelldomäne kann z.B. aus Bildern von handgeschriebene Ziffern und die Zieldomäne aus Bildern von Hausnummernschildern bestehen. Die Ausgangs- und Zielaufgabe ist jeweils die Klassifikation der Ziffern.
Formal [PY09] ist eine Domäne $\mathcal{D}$ und eine Aufgabe $\mathcal{T}$ definiert als:

$$\mathcal{D} = \{\mathcal{X}, p(\mathbf{x})\} \tag{2.26}$$

$$\mathcal{T} = \{\mathcal{Y}, p(\mathbf{y}|\mathbf{x})\} \tag{2.27}$$

wobei $\mathcal{X}$ der gesamte Merkmalsraum (z.B. alle 256×256 RGB-Pixelwerte) und $\mathcal{Y}$ der gesamte Labelraum (z.B. alle bekannten Tierarten) ist. $p(\mathbf{x})$ beschreibt die Verteilung der Trainingsdaten mit

$$X = \{x_i \in \mathcal{X} | 0 \leq i \leq n\} \tag{2.28}$$

und $Range(\mathbf{x}) = X$. $p(\mathbf{y}|\mathbf{x})$ ist die bedingte Verteilung der Labels in Abhängigkeit der Eingabedaten, welche typischerweise durch ein Modell $p(\mathbf{x}|\mathbf{y}, \theta)$ approximiert wird. Die Parameter θ werden dabei auf Basis eines Datensatzes D geschätzt.

$$D = \{(x_i, y_i) \in X \times \mathcal{Y} \mid 0 \leq i \leq n\} \tag{2.29}$$

Ausgehend von einer Quelldomäne $\mathcal{D}_S$, einer entsprechenden Ausgangsaufgabe $\mathcal{T}_S$ sowie einer Zieldomäne $\mathcal{D}_T$ und einer Zielaufgabe $\mathcal{T}_T$ besteht das Ziel des Transferlernens nun darin, die bedingte Verteilung $p(\mathbf{y}_T|\mathbf{x}_T)$ aus $\mathcal{D}_T$ mit dem aus $\mathcal{D}_S$ und $\mathcal{T}_S$ gewonnenem Wissen zu lernen, wobei $\mathcal{D}_S \neq \mathcal{D}_T$ oder $\mathcal{T}_S \neq \mathcal{T}_T$. Daraus ergeben sich die folgenden Szenarien:

- $\mathcal{X}_S \neq \mathcal{X}_T$
  Quell- und Zieldomäne haben unterschiedliche Merkmalsräume, z.B. Texte in verschiedenen Sprachen oder Schwarz-Weiß-Bilder in der Quelldomäne und Farbbilder in der Zieldomäne.

- $p(\mathbf{x}_S) \neq p(\mathbf{x}_T)$
  Die Verteilung der Daten unterscheidet sich, z.B. Texte in der gleichen Sprache, aber mit unterschiedlichem thematischen Schwerpunkten oder Musikstücke aus unterschiedlichen Genres.

- $\mathcal{Y}_S \neq \mathcal{Y}_T$
  Die Labelräume zwischen Ausgangs- und Zielaufgabe stimmen nicht überein, z.B. wenn die Zieldomäne mehr oder gar keine Klassen aufweist.

- $p(\mathbf{y}_S|\mathbf{x}_S) \neq p(\mathbf{y}_T|\mathbf{x}_T)$
  Die bedingten Verteilungen unterscheiden sich, z.B. wenn die Häufigkeit der Klassen in der Quell- und Zielverteilung sehr unterschiedlich ist.

Unterscheiden sich Ausgangs- und Zielaufgabe, so wird von *induktivem Transferlernen* gesprochen, während beim *transduktiven Transferlernen* Ausgangs- und Zielaufgabe übereinstimmen.

2.2 Verarbeitung akustischer Signale

Das Hören gehört zu den fünf menschlichen Sinnen und erlaubt es uns beispielsweise zu musizieren oder miteinander zu kommunizieren. Akustische Signale spielen also eine wichtige Rolle im alltäglichen Leben und stellen ein physikalisches Phänomen dar, welches mit Hilfe mathematischer Werkzeuge untersucht werden kann. Dazu wird in Abschnitt 2.2.1 zunächst näher auf den physikalischen Prozess der Tonentstehung und die wichtigsten Kenngrößen eingegangen. Anschließend wird in Abschnitt 2.2.2 und Abschnitt 2.2.3 besprochen, wie Ton in seine Spektraldarstellung überführt werden kann, um so eine kompakte Übersicht über die verschiedenen Frequenzen im Ton zu erhalten.

2.2.1 Entstehung von Tönen

Jedem Ton und jedem Geräusch geht die Vibration eines Objekts, wie einer Gitarrensaite oder den Stimmbändern, voraus. Damit sich der Ton ausbreiten kann, benötigt er ein geeignetes Ausbreitungsmedium. Ton kann sich in Feststoffen, Flüssigkeiten und Gasen ausbreiten, nicht jedoch in einem Vakuum. Folglich herrscht im Weltraum völlige Stille.
Wenn ein Objekt vibriert, versetzt es die der Vibrationsquelle am nächsten liegenden Teilchen des Mediums in Schwingung. Die Teilchen werden durch die intramolekularen Kräfte zusammengehalten, die es ihnen lediglich erlauben, um ihre Position zu oszillieren. Durch die Vibration werden sie aus ihrer Ruheposition (Equilibrium) gedrängt und beginnen mit den benachbarten Teilchen zu kollidieren, welche daraufhin wiederum in Schwingung versetzt werden. Dieser Prozess setzt sich fort, bis der Schall das Ohr erreicht. Die Vibrationen werden über das Trommelfell an die Cochlea (Hörschnecke) im Innenohr weitergegeben, wo diese in elektrische Impulse umgewandelt und an das Gehirn weitergeleitet werden.
Töne breiten sich im Medium also in Form von Druckschwankungen (Kompressionen und Expansionen) aus. Die Oszillation von Druckzuständen wird gemeinhin als Schallwelle bezeichnet. Abbildung 2.10 stellt diesen Sachverhalt visuell dar. Die *Frequenz* ist die Geschwindigkeit, mit der sich Kompression und Expansion abwechseln. Sie wird in Oszillationen pro Sekunde (*Hertz*, Hz) gemessen. Je höher die Frequenz, desto höher der wahrgenommene Ton [AM10]. Die Frequenzen im hörbaren Bereich liegen dabei zwischen 20Hz und 20kHz.

Die Lautstärke (akustische Energie) hängt von der Anzahl der beteiligten Partikel und somit dem Schalldruck ab. Die *Amplitude* der Schwallwelle repräsentiert demnach den Schalldruck, der von einer Schallwelle ausgeübt wird. Dafür wird der Druck p_{m}, gemessen in Pascal (Pa), mit die Hörschwelle (geringste wahrnehmbare Lautstärke) $p_0 = 2 * 10^{-5} Pa = 20\mu Pa$ wie folgt ins Verhältnis gesetzt:

$$10 * \log_{10}\left((\frac{p_m}{p_0})^2\right)\mathrm{dB} \tag{2.30}$$

Der Quotient beschreibt das Verhältnis des gemessenen Drucks zur Hörschwelle. Die Quadrierung dient der Abbildung der Schallintensität, der Kraft, die pro Flächeneinheit ausgeübt wird. Da sich die Schallintensitäten, welche der Mensch wahrnehmen kann, über viele Zehnerpotenzen erstrecken, wird der dekadische Logarithmus angewandt, um den Wert in die logarithmische Skala zu konvertieren. Der Faktor 10 wird multipliziert, um den Wert für einen Menschen besser interpretierbar zu machen und von der Einheit Bel in *Dezibel* (dB) zu wechseln. Folglich bedeutet eine Verdopplung der Lautstärke eine Änderung von +10dB. Eine Unterhaltung liegt indes bei etwa 40dB, ein vorbeifahrender Zug bei 80dB und die Schmerzgrenze bei 130dB.
Um Töne in ein, vom Computer verarbeitbares Format zu überführen, müssen die Druckschwankungen mit Hilfe eines Umformers (etwa einem Mikrofon) zunächst in eine Sequenz von elektrischen Impulsen übersetzt werden. Bei der Digitalisierung wird dieses Analogsignals in ein zeit- und wertdiskretes Signal umgewandelt. Die *Samplingrate* (Abtastrate) bestimmt dabei, wie häufig das Analogsignal pro Sekunde in gleichmäßigen Abständen abgetastet wird. Sie muss gemäß dem *Nyquist-Shannon-Abtasttheorem* [Sha49] mehr als doppelt so hoch sein wie die höchste Frequenz des abzutastenden Signals. Ansonsten entstehen Artefakte, die im Ausgangssignal nicht vorhanden waren.
Zuletzt gibt die *Bittiefe* die Anzahl der Bits an, die bei der Abtastung pro Abtastwert verwendet werden. Je mehr Bits zur Verfügung stehen, desto genauer sind die Abtastwerte.

2.2.2 Diskrete Fourier-Transformation

Die Fourier-Transformation ist einer der Meilensteine der angewandten Mathematik und ermöglicht unter anderem die Bild- und Audiokompression in Echtzeit, das Lösen von partiellen Differentialgleichungen oder die Rauschunterdrückung in Signalen.
Fourier's Theorem besagt, dass jede periodische Funktion als unendliche Linearkombination von Sinus- und Kosinustermen, der *Fourier-Reihe*, ausgedrückt werden kann [BB86]. Präziser formuliert ist jede periodische und abschnittsweise stetige Funktion $f(t)$, definiert in $[-T, T)$, durch die komplexe Fourier-Reihe darstellbar:

$$f(t) = \sum_{n=-\infty}^{\infty} (a_n + ib_n) * \left(\cos(\frac{\pi nt}{T}) + i\sin(\frac{\pi nt}{T})\right) = \sum_{n=-\infty}^{\infty} c_n * \exp(i\frac{\pi nt}{T}) \tag{2.31}$$

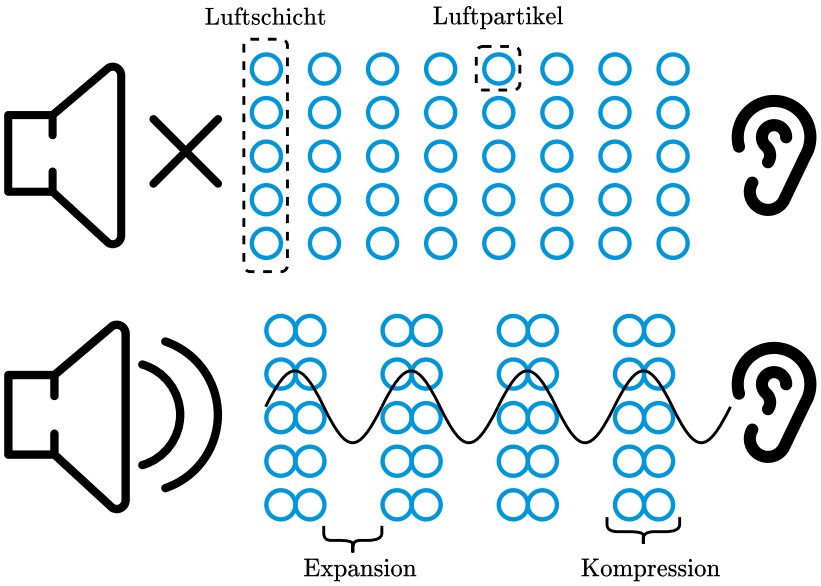


Abbildung 2.10: Veranschaulichung der Ausbreitung von Schallwellen. Die Expansion und Kompression der Luftpartikel führt zu Druckschwankungen, welche sich von der Quelle bis zum Trommelfell ausbreiten.

Gesucht sind nun alle Koeffizienten c_k.
Da die Druckschwankungen (Töne) in Abhängigkeit der Zeit zwar eine Wellenform annehmen, jedoch in den häufigsten Fällen nicht periodisch sind, eignet sich die Fourier-Reihe nur bedingt zur Beschreibung, Analyse und Zerlegung von Tönen.
Um die idealisierte Annahme der Periodizität zu umgehen, fasst die *Fourier-Transformation* eine nicht-periodische Funktion als periodische Funktion mit unendlicher Periode auf. Sie ist im Wesentlichen der Grenzwert der Fourier-Reihe, wenn die Länge des Definitionsbereichs gegen unendlich strebt [BK19]. Sie lässt sich einsetzen, um ein Signal im Zeitspektrum in das *Frequenzspektrum* zu transformieren, vgl. Abbildung 2.11. Das Frequenzspektrum ordnet jeder Frequenz die entsprechende Amplitude zu. Dieser Sachverhalt wird in Abbildung 2.11 schematisch verdeutlicht.
Im Kontext der Verarbeitung akustischer Signale lässt sich daher ein Geräusch, welches aus Tönen verschiedener Frequenzen besteht, mit Hilfe der Fourier-Transformation in seine Einzeltöne unter Angabe ihrer Frequenz und Amplitude zerlegen.

Die Fourier-Reihe und die Fourier-Transformation sind für stetige Funktionen definiert. Wird mit Realweltdaten gearbeitet, so liegen diese jedoch nur in Form von einer endlichen Menge an Messungen von $f(t)$ vor. Daraus resultiert die Notwendigkeit, die Fourier-Transformation für diskrete, vektorisierte Datenformate zu approximieren.

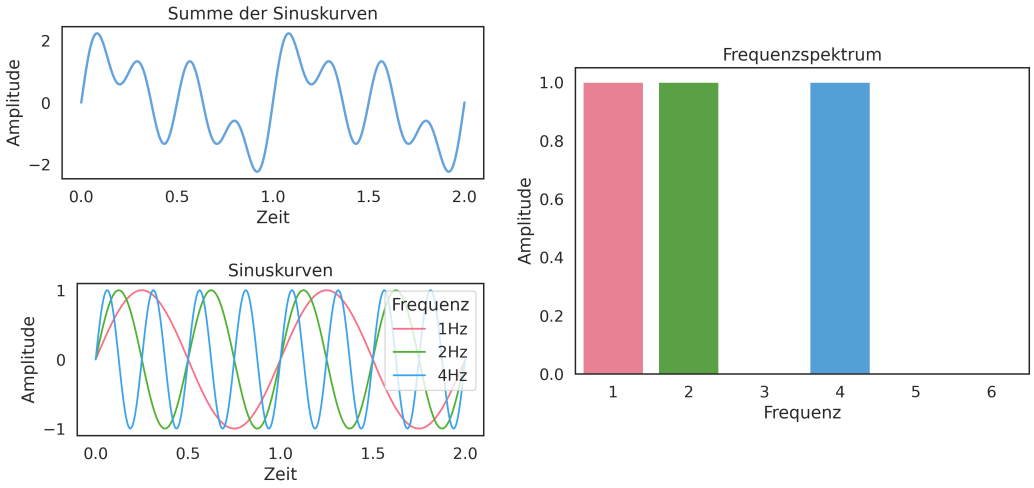


Abbildung 2.11: Exemplarische Veranschaulichung der Zerlegung eines Signals im Zeitspektrum (links oben) in seine Bestandteile (links unten) sowie dessen Transformation in das Frequenzspektrum (recht). Das Frequenzspektrum ordnet jeder Frequenz die entsprechende Amplitude zu.

Dazu sei f der Vektor von n Funktionswerten, die in gleichmäßigen Abständen ermittelt wurden, und $\hat{f}$ der Vektor der gesuchten Koeffizienten (Amplituden) zu den Frequenzen $j = 0 \dots n-1$:

$$f = \begin{bmatrix} f_0 \\ \vdots \\ f_{n-1} \end{bmatrix} \qquad \hat{f} = \begin{bmatrix} \hat{f}_0 \\ \vdots \\ \hat{f}_{n-1} \end{bmatrix} \tag{2.32}$$

Die *diskrete Fourier-Transformation* berechnet die Koeffizienten nun wie folgt:

$$\hat{f}_k = \sum_{j=0}^{n-1} f_j * \exp(\frac{-i2\pi jk}{n}) \tag{2.33}$$

Eine effizientere und kompaktere Darstellung wird durch die Verwendung der Matrixschreibweise erreicht:

$$\hat{f} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n^1 & \omega_n^2 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \cdots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \cdots & \omega_n^{(n-1)^2} \end{bmatrix} f, \quad \text{mit } \omega_n = \exp(\frac{-i2\pi}{n}) \tag{2.34}$$

Da die Anwendung der Matrix $\mathcal{O}(n^2)$ Operationen benötigt, wird in der Praxis die *Fast-Fourier-Transformation* verwendet. Durch sie lässt sich die Anzahl der

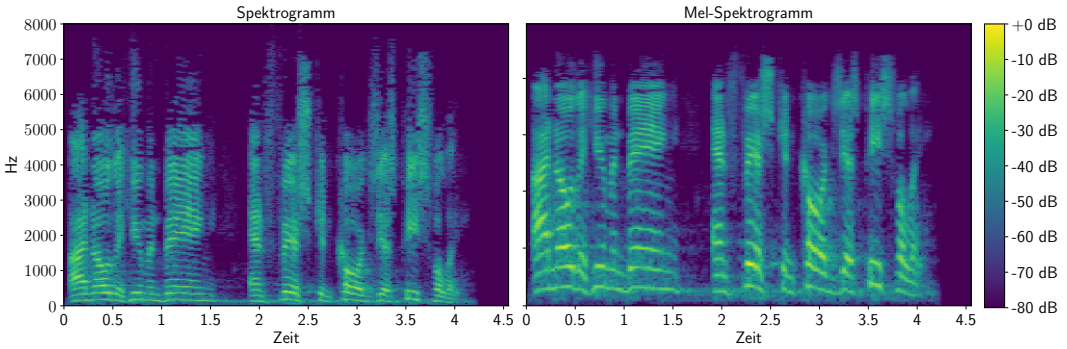


Abbildung 2.12: Vergleich eines Spektrogramms mit einem Mel-Spektrogramm. Ein Spektrogramm zeigt an, wie ausgeprägt die einzelnen Frequenzen zu einem bestimmten Zeitpunkt vorhanden sind. Beim Mel-Spektrogramm werden Unterschiede zwischen niedrigeren Frequenzen verstärkt, Unterschiede zwischen höheren Frequenzen abgeschwächt.

Operationen auf $\mathcal{O}(n \log(n))$ reduzieren.

2.2.3 Mel-Spektrogramm

Ein Nachteil der Darstellung eines Signals im Frequenzspektrum ist, dass es die Frequenzanteile über den gesamten zeitlichen Verlauf des zugrundeliegenden Signals angibt. Die Veränderung des Frequenzspektrums über die Zeit ist folglich nicht dargestellt. Das Frequenzspektrum einer aufgenommenen Konversation zwischen zwei Personen lässt z.B. nur Rückschlüsse auf deren Stimmlage zu. Wann welche Person welchen Laut (Phonem) von sich gegeben hat, wird nicht ersichtlich. So wird deutlich, dass in vielen Fällen eine Darstellung des Frequenzspektrums eines Signals in Abhängigkeit der Zeit erforderlich ist. Eine solche Darstellungsform nennt sich *Spektrogramm* und ist auf der linken Seite von Abbildung 2.12 dargestellt.
Zur Erstellung des Spektrogramms wird die *Short-Time Fourier Transformation* verwendet. Anstatt die Fourier-Transformation über den gesamten zeitlichen Bereich zu berechnen, wird ein kleines überlappendes (Gauß-)Fenster fester Größe über das Ausgangssignal geschoben und die Fourier-Transformation auf den aktuellen Fensterinhalt angewandt. Daraus ergibt sich eine Sequenz von Frequenzspektren. Aus der Konkatenation dieser entsteht schließlich das Spektrogramm. Die Auflösung, d.h. die Zeitspanne, über die ein einzelnes Frequenzspektrum im Spektrogramm berechnet wurde, hängt dabei von der Schrittlänge und der Fenstergröße ab. Darüber hinaus stellt es den Frequenzbereich linear dar. Der Mensch nimmt die Tonhöhe (Frequenz) jedoch logarithmisch wahr. Er kann Unterschiede in niedrigeren Frequenzen wesentlich besser erkennen als in höheren Frequenzen. Obwohl der Unterschied zwischen 100Hz und 300Hz und 10.000 Hz und 10.200Hz, jeweils genau 200 Hz beträgt, nimmt

der Mensch ersteren deutlich stärker wahr.
Auf Basis dieser Beobachtung entwickelten 1937 Stevens et al. [SVN37] die *Mel-Skala* (Abkürzung für Melodie-Skala), bei der gleiche Abstände in der Tonhöhe für den Hörer gleich weit entfernt klingen.
Um einen Frequenzwert von f Hz in die Mel-Skala zu konvertieren, wurde folgende Formel empirisch durch psychoaukustische Versuche ermittelt:

$$m = 2595 * \log(1 + \frac{f}{700}) \tag{2.35}$$

Um das Mel-Spektrogramm zu erstellen, muss zunächst m für die minimale und maximale Frequenz berechnet werden. Danach werden n gleichmäßig verteilte Punkte in $[f_{min}, f_{max}]$ bestimmt und zurück in Hz konvertiert.

$$f = 700 * (10^{\frac{2595}{m}} - 1) \tag{2.36}$$

Das Resultat sind die Zentren der n *Mel-Bänder*. Ein Mel-Band fasst alle Frequenzen in einem gewissen Intervall zusammen. Jede Frequenz wird dem ihr am nächsten liegenden Mel-Band zugeordnet. Durch die Mel-Skalierung werden den oberen Mel-Bändern mehr Frequenzen zugeordnet als den unteren, was der logarithmischen Wahrnehmung der Tonhöhe Rechnung trägt. Abbildung 2.12 veranschaulicht den Unterschied zwischen einem Spektrogramm und einem Mel-Spektrogramm.

2.3 Fazit und Verwendung der Grundlagen

In diesem Kapitel wurden Definitionen und Begrifflichkeiten eingeführt, welche in den anschließenden Kapiteln der vorliegenden Arbeit zur Anwendung kommen. Dazu wurden zunächst die Grundbegriffe und aktuellen Anwendungen des maschinellen Lernens besprochen. Des Weiteren wurden die mathematischen Grundlagen für die Schätzung von Modellparametern sowie die Modellklasse der neuronalen Netze eingeführt. Überdies wurde darauf eingegangen, wie neuronale Netze zur Merkmalsextraktion und zum Wissenstransfer genutzt werden können. Schließlich wurde der physikalische Prozess der Tonentstehung näher beleuchtet und besprochen, wie Ton in seine Frequenzbestandteile zerlegt werden kann.
In den nachfolgenden Kapiteln werden die vorgestellten Grundlagen wie folgt angewandt: Kapitel 3 verwendet rekurrente neuronale Netze (Abschnitt 2.1.6), um geeignete Repräsentationen (Abschnitt 2.1.7) zur Klassifizierung und der Erkennung von Anomalien in den Spektraldarstellungen (Abschnitt 2.11) von Geräuschen zu lernen (Abschnitt 2.1.3).
Kapitel 4 untersucht die Auswirkungen des Wissenstransfers (Abschnitt 2.1.7) von vortrainierten CNNs (Abschnitt 4.1.3.1) auf die akustische Anomalieerkennung. Schließlich widmet sich Kapitel 5 Anwendungen des Repräsentationslernens (Abschnitt 2.1.7), die über die akustische Signalverarbeitung hinausge-

hen. Eine genauere Diskussion der Grundlagen des Reinforcement Learnings und des Clusterings ist in Abschnitt 5.1.1 zu finden, da diese nur für den in Abschnitt 5.1 vorgestellten Ansatz relevant sind.

3 Repräsentationslernen für akustische Daten

Dieses Kapitel widmet sich der Verarbeitung akustischer Daten mit Hilfe von neuronalen Netzen. Dabei wird die Fragestellung untersucht, wie neuronale Repräsentationen des Mel-Spektrogramms gelernt werden können, die für das angedachte Anwendungsszenario möglichst gut geeignet sind.
Die Abschnitte 3.1, 3.2 und 3.3 sind dabei strikt nach dem jeweiligen Anwendungsfall getrennt und basieren in gleicher Reihenfolge auf den Publikationen [MIL21a; Mül+21d; MIL21b].
Zunächst wird in Abschnitt 3.1 eine rekurrente neuronale Netzarchitektur zur Klassifikation von Vokalisationen von Primaten vorgeschlagen. Da in diesem Fall ein komplett annotierter Datensatz vorliegt, ist der vorgestellte Ansatz dem Supervised Learning zuzuordnen. Aufgrund der guten Klassifikationsleistung der Architektur eignet sie sich besonders für die automatische Überwachung von Wildtieren und kann folglich genutzt werden, um den Rückgang der Biodiversität und der Artenvielfalt zu bekämpfen.
Der darauffolgende Abschnitt 3.2 behandelt die akustische Leckerkennung in Wasserversorgungsnetzen. Da zum Trainingszeitpunkt lediglich Aufnahmen des Normalbetriebs verfügbar sind, ist er dem Unsupervised Learning zuzuordnen. Die akustischen Aufnahmen wurden mit Körperschallmikrofonen entlang eines Wasserversogungsnetzes aufgezeichnet und werden im folgenden Abschnitt verwendet, um mit neuronalen Netzen und klassischen Feature-Engineering-Verfahren ein Modell des Normalfalls zu lernen. Das vorgestellte Verfahren kann die negativen Auswirkungen einer Leckage auf die umliegende Infrastruktur reduzieren und die Reparaturkosten senken.
Generell ist die Erkennung anomaler Geräusche ein wichtiger und bedeutender Anwendungsbereich der künstlichen Intelligenz in der Industrie. Abschnitt 3.3 beschäftigt sich deshalb mit der Erkennung anomaler Geräusche in industriellen Fertigungsanlagen. Zu diesem Zweck wird, ähnlich zu Abschnitt 3.1, eine rekurrente Netzarchitektur vorgestellt. Indem Teile des Mel-Spektrogramms entfernt und als Vorhersageziel des Netzes definiert werden, kann der Vorhersagefehler zur Anomaliebewertung genutzt werden.
Abschnitt 3.4 fasst das Kapitel und dessen wichtigste Erkenntnisse kurz zusammen. Eine detailliertere Zusammenfassung der vorgestellten Algorithmen sowie ein Ausblick auf mögliche Ansatzpunkte für zukünftige Arbeiten, ist am Ende eines jeden Abschnitts zu finden.

3.1 Akustische Klassifikation von Primaten

Um den Rückgang der Biodiversität und Artenvielfalt aufzuhalten, sind präzise und zuverlässige Instrumente zur Überwachung von Wildtieren unerlässlich. Die Überwachung von Wildtieren ist darüber hinaus ein wesentlicher Bestandteil der meisten Artenschutzmaßnahmen und wird zur Verfolgung von Bewegungsmustern, der Populationsdemografie und der Sozialdynamik sowie zur Seuchenbekämpfung und zur Verhinderung von Wilderei eingesetzt. Die Gewinnung dieser Informationen ist besonders wichtig, um gefährdete Arten zu schützen. Generell ermöglicht die Überwachung Einblicke in die räumlichen und zeitlichen Zusammenhänge, in denen Individuen und Populationen miteinander interagieren [Blu+11; Hei+15].
Einer der zahlreichen Bausteine für das Wildtiermonitoring ist die akustische Überwachung. Im Vergleich zur visuellen Überwachung hat diese den Vorteil, dass sie auch in Gebieten mit hoher Vegetation (z. B. im dichtem Regenwald) eingesetzt werden kann und in der Regel kostengünstiger ist. Ein praxistaugliches Wildtierüberwachungssystem kombiniert dabei Messungen von verschiedenen Sensoren wie Kameras, autonomen Aufnahmegeräten und Bewegungsmeldern, um die Aussagekraft und Zuverlässigkeit der gewonnenen Informationen zu maximieren. Schwächen in einem dieser Teilsysteme führen zu einer Verschlechterung der Gesamtleistung des Systems.
In dieser Arbeit liegt der Schwerpunkt auf der Klassifizierung von Vokalisationen (Lauten) von Primaten, eine Aufgabe, die sich gut in das oben beschriebene Gesamtbild einfügt.
In früheren Arbeiten wurden die akustische Klassifizierung von Primaten und verwandte Aufgaben durch die Extraktion von Mel Frequency Cepstral Coefficients und deren Verwendung als Input für traditionelle Klassifizierer wie Support-Vector Machines [Hei+15; FZD16; CCM19] und Multi-Layer Perceptrons [MZ13] gelöst. Während tiefe neuronale Netze in der akustischen Signalverarbeitung allgegenwärtig sind, haben diese im Zusammenhang mit der akustischen Klassifikation von Primaten bis jetzt nur wenig Aufmerksamkeit erhalten.
Folglich wird in diesem Abschnitt ein Deep-Learning-Ansatz auf der Basis bidirektionaler rekurrenter neuronaler Netze vorgeschlagen. Der Ansatz imitiert die traditionelle Merkmalsextraktion aus Mel-Spektrogrammen, ist aber vollständig differenzierbar. Folglich extrahiert der Ansatz automatisiert Repräsentationen, welche für die vorliegende Aufgabe geeignet sind. Zusätzlich werden auch Aufnahmen mit variabler Länge nativ unterstützt. Zu diesem Zwecke werden mehrere bewährte Module wie Pooling, Normalized Softmax [ZW18] und Focal Loss [Lin+17] zu einem neuen Ansatz für die Primatenklassifikation kombiniert.
Da der Ansatz eine Vielzahl von Hyperparametern erfordert, werden diese nicht manuell, sondern mittels Bayes'scher Hyperparameter-Optimierung [Ber+11] ermittelt.

Die Methode anhand eines kürzlich veröffentlichten Datensatz zur Primatenklassifikation getestet. Hierbei wird ein *Unweighted Average Recall* (UAR) von $89,4\%$ auf dem Validierungsdatensatz und $89,3\%$ auf dem Testdatensatz erreicht, wobei ein Ensemble der fünf leistungsfähigsten Modelle verwendet wird, die während der Hyperparameteroptimierung ermittelt wurden. Der Ansatz übertrifft die beste Baseline von [Sch+21], die eine UAR von 87.5% auf dem Testset erreicht.

3.1.1 Bestehende Ansätze zur Klassifikation von Tiergeräuschen

In früheren Studien wurde bereits die automatische akustische Überwachung von Walen [BD13], Vögeln [PMC18], Fledermäusen [RV16], Insekten [GP07], Amphibien [Bra+16] und Elefanten [Wre+17] untersucht. Darüber hinaus beschäftigen sich einige Arbeiten explizit mit Klassifikationsproblemen auf Basis von Primatenvokalisationen. Typische Problemstellungen sind dabei die Erkennung der Rufart [MZ13; Tur+16], der Spezies [Hei+15; MZ13], des Alters [FZD16] oder der Unterscheidung von Individuen [CCM19].
Alle genannten Ansätze haben jedoch die Gemeinsamkeit, dass sie auf manuellem Feature Engineering beruhen und daher die Flexibilität und Leistungsfähigkeit neuronaler Netze nicht ausgenutzt werden. Darüber hinaus vernachlässigen diese Ansätze die zeitliche Dimension der Vokalisationen, da die Merkmale durch voneinander unabhängige gleitende Fenster extrahiert werden.
Im Gegensatz dazu wird in dieser Arbeit eine rekurrente neuronale Netzarchitektur vorgeschlagen und gezeigt, dass diese den klassischen Ansätzen überlegen ist.
Ein weit verbreiteter Ansatz zur akustischen Klassifikation basiert auf der Verwendung von CNNs [Her+17a; Kon+20; VBV22] Auch wenn in der Praxis durch die Anwendung von CNNs gute Ergebnisse erzielt werden können, passen RNNs konzeptionell besser zu den Eingabedaten, da Mel-Spektrogramme als Sequenz von Spaltenvektoren angesehen werden können. CNNs hingegen wurden ursprünglich nicht für die Modellierung von temporalen, sondern nur von räumlichen Zusammenhängen entwickelt.
Ansätze, die ebenfalls auf RNNs basieren [Dai+16; Pha+17; SS19], sind zum einen für andere Anwendungszwecke wie der Klassifikation von Musik-Genres oder Umweltgeräuschen ausgelegt, zum anderen verwenden sie einfachere Pooling-Mechanismen und Trainingsziele als der hier vorgestellte Ansatz.
In letzter Zeit haben sich Transformer [Vas+17], welche in hohem Maße auf dem Attention-Mechanismus basieren, als äußert leistungsfähige Alternative erwiesen [GCG21; Nag+21; Che+22]. Da diese aber stärker von großen Datensätzen abhängig sind, deren Training komplizierter ist und eine größere Anzahl an Modellparametern erfordern [Dos+21], werden in dieser Arbeit RNNs bevorzugt.

3.1.2 Eine rekurrente Netzarchitektur zur akustischen Klassifikation

Vor dem Aufkommen des Deep Learnings wurden Merkmale des zugrunde liegenden Signals durch das Feature Engineering manuell extrahiert. Hierzu war häufig Fachwissen von Domänenexperten nötig. Damit diese Merkmale aussagekräftig und informativ bleiben und somit die lokalen Charakteristiken erhalten bleiben, wurden sie üblicherweise mit einem kleinen gleitenden Fenster über das Signal extrahiert. Zur Beschreibung des gesamten Signals wurden die extrahierten Merkmalsvektoren zu einigen wenigen statistischen Messwerten wie Mittelwert, Median und Standardabweichung oder den Mindest- und Höchstwerten der Merkmale aggregiert. Tatsächlich wird dieses Verfahren auch heutzutage noch angewandt und eignet sich als konkurrenzfähige Baseline [EWS10; Zwe+21].
Der vorliegende Ansatz lehnt sich an dem beschriebenen Arbeitsablauf an, zielt aber darauf ab, diesen vollständig differenzierbar zu machen, sodass die Leistungsfähigkeit neuronaler Netze (NNs) genutzt werden kann. Durch die Nutzung von NNs verschiebt sich der Schwerpunkt von der manuellen Suche nach geeigneten Merkmalen hin zur Suche nach einer geeigneten Netzarchitektur und den dazugehörigen Hyperparametern.
Im Rahmen dieser Arbeit wird davon ausgegangen, dass Audiosignale bereits in ihrer Zeit-Frequenz-Darstellung (hier das Mel-Spektrogramm) vorliegen. Demzufolge ist der Trainingsdatensatz $\mathcal{D}$, der n Mel-Spektrogramme mit variabler Länge enthält, wie folgt gegeben:

$$\mathcal{D} = \{(x_i, y_i) \in \mathbb{R}^{F \times T_i} \times \mathbb{N} \mid 1 \leq i \leq n\} \tag{3.1}$$

Dabei ist F die Anzahl der Frequenzbänder, T_i die Anzahl der Zeitfenster und y_i eine ganze Zahl, die das Label angibt. Die Variabilität der Länge jedes Mel-Spektrogramms x_i ist durch die unterschiedliche Anzahl von Zeitfenstern T_i angegeben. Diese Eigenschaft ist in der Praxis sehr häufig anzutreffen, weshalb in dieser Arbeit ein Modell vorgestellt wird, das Mel-Spektrogramme variabler Länge nativ unterstützt. Dadurch entfällt die Frage, wie das Mel-Spektrogramm in gleich große Teile zerlegt werden soll. Die Frage, wie die Vorhersagen für die einzelnen Teilstücke wieder zusammengefasst werden soll, muss ebenfalls nicht mehr beantwortet werden.
Es ist zu beachten, dass jedes x_i als eine Folge von T_i Spaltenvektoren mit einer Dimensionalität von F interpretiert wird.
Unter dieser Interpretation können NNs verwendet werden, die speziell für die Verarbeitung von sequentiellen Daten entwickelt wurden. Die bekanntesten Vertreter dieser Art von NNs sind LSTMs [HS97] und Transformers [Vas+17]. Im vorliegenden Fall erwiesen sich LSTMs als einfacher und stabiler zu trainieren, weshalb im Folgenden ausschließlich LSTMs zum Einsatz kommen.
Hier werden LSTMs verwendet, um kontextualisierte Repräsentationen für jeden Eingabevektor (jedes Zeitfenster) zu berechnen. Um die Länge der Se-

quenz zu reduzieren und damit auch die Anzahl der Zeitschritte, durch die der Gradient zurückpropagiert werden muss, zu verringern, werden zwei aufeinander folgende Zeitfenster des Mel-Spektrogramms konkateniert. Durch diese Vorgehensweise halbiert sich die Sequenzlänge und verdoppelt sich die Merkmalsdimension. Da dies nur für Mel-Spektrogramme mit einer geraden Anzahl von Zeitrahmen möglich ist, wird bei einer ungeraden Anzahl von Zeitrahmen der letzte Zeitrahmen des Mel-Spektrogramms wiederholt. Dieses Vorgehen ist ebenfalls inspiriert durch die manuelle Merkmalsextraktion, bei der Merkmale über mehrere Zeitschritte, aber nicht über die gesamte Länge berechnet werden.
Die Funktion squash realisiert die beschriebene Stauchung eines Mel-Spektrogramms $M \in \mathbb{R}^{F \times T}$.

$$\text{squash} : \mathbb{R}^{F \times T} \to \mathbb{R}^{2F \times \frac{T}{2}}, \text{ wobei } \frac{T}{2} \in \mathbb{N} \tag{3.2}$$

$$\text{squash}\left(\begin{bmatrix} | & | & | \\ v_1 & \cdots & v_T \\ | & | & | \end{bmatrix}\right) = \begin{bmatrix} | & | & | \\ v_1 & \cdots & v_{(T/2)-1} \\ | & | & | \\ | & | & | \\ v_2 & \cdots & v_{(T/2)} \\ | & | & | \end{bmatrix} = \begin{bmatrix} | & | & | \\ \hat{v_1} & \cdots & \hat{v}_{(T/2)} \\ | & | & | \end{bmatrix} \tag{3.3}$$

Ein bidirektionales LSTM (BiLSTM) berechnet anschließend für jedes Zeitfenster $\hat{v_i}$ einen versteckten Zustand. Das BiLSTM verkettet dabei schlicht die H-dimensionalen verborgenen Zustände, die entstehen, wenn squash(M) in normaler und umgekehrter Reihenfolge durch das LSTM geleitet wird. So entstehen $\frac{T_i}{2}$ (Many-to-many) hoch kontextualisierte Repräsentationen $\in \mathbb{R}^{2H}$, d.h. jede Repräsentation beinhaltet Informationen über die verborgenen Zustände aller anderen Zeitfenster. Somit gilt:

$$\text{BiLSTM} : \mathbb{R}^{2F \times (T/2)} \to \mathbb{R}^{2H \times (T/2)} \tag{3.4}$$

Um die Anzahl der gewonnenen Repräsentationen zu reduzieren und kompakt zusammenzufassen, werden diese im letzten Schritt durch Min-, Max- und Mean-Pooling über die zeitliche Dimension aggregiert. Die endgültige Repräsentation m für M ist dann durch die Konkatenation der Pooling-Ergebnisse gegeben.

$$\hat{M} = \text{BiLSTM}(\text{squash}(M)) \tag{3.5}$$

$$m = \begin{bmatrix} \text{mean}(\hat{M}) \\ \min(\hat{M}) \\ \max(\hat{M}) \end{bmatrix}, \quad m \in \mathbb{R}^{3*2*H} \tag{3.6}$$

Dieses Vorgehen erfolgt ebenfalls in Anlehnung an das Paradigma der Feature-Extraktion. Die Pooling-Operationen wurden aufgrund ihrer Einfachheit gewählt. Künftige Arbeiten könnten sich mit komplexeren Varianten auseinandersetzen.
Zur Vorhersage der Klassenzugehörigkeit kann ein FFNN verwendet werden, welches m als Eingabe entgegennimmt.
Der gängigste Ansatz für das Training eines neuronalen Klassifizierers ist die Verwendung der *Softmax*-Aktivierungsfunktion in der letzten Schicht des FFNN, um die Aktivierungen (*Logits*) in Wahrscheinlichkeiten umzuwandeln.
Seien $W \in \mathbb{R}^{D \times C}$ und $b \in \mathbb{R}^D$ die Gewichte der letzten linearen Schicht eines FFNN $\phi : \mathcal{X} \to \mathbb{R}^D$ um anhand der Aktivierungen eines Trainingsbeispiels $\phi(x_i),\ x_i \in \mathcal{X}$ der letzten versteckten Schicht der Dimension D die Vorhersage für die C verschiedenen Klassen zu berechnen. Mit Hilfe des Softmax lassen sich dann die Parameter μ einer kategorischen Wahrscheinlichkeitsverteilung $\mathrm{Cat}(\mathbf{c}|\mu)$ über die C Klassen berechnen.

$$\mu(x_i) = \frac{\exp(\phi(x_i)^T W + b)}{\Sigma \exp(\phi(x_i)^T W + b)} \tag{3.7}$$

$$\mathrm{Cat}(c|\mu(x_i)) = \mu(x_i)_c \tag{3.8}$$

Wobei $\sum_{c=1}^{C} \mu_c = 1$.
Der Softmax-Klassifikator verfügt jedoch über Schwächen bei der Modellierung schwieriger oder strukturell ungewöhnlicher Trainingsbeispiele. Die Erhöhung der $L2$-Norm $\|\phi(x_i)\|_2$ von einfach zu klassifizierenden Beispielen genügt für dessen Minimierung [RCC17; Wan+17; ZW18]. Im Falle der Klassifikation von Primatenvokalisationen variieren beispielsweise die Lautstärke, die Dauer und die Anzahl der Beispiele pro Primat sehr stark. Damit laute, gut hörbare Vokalisationen von Primaten mit hoher Populationsdichte das Training nicht dominieren, wird die Softmax Aktivierungsfunktion wie folgt angepasst:

$$\mu(x_i) = \frac{\exp\left(\phi(x_i)^T W / (\tau * \|\phi(x_i)^T\| * \|W\|)\right)}{\Sigma \exp\left(\phi(x_i)^T W / (\tau * \|\phi(x_i)^T\| * \|W\|)\right)} \tag{3.9}$$

Durch den *Normalized Softmax* (Gleichung 3.9) werden anstelle der unbeschränkten Skalarprodukte $\phi(x_i)^T W$ die Kosinus-Ähnlichkeiten berechnet, sodass $\phi(x_i)$ und W auf die Hypersphäre projeziert werden und $\tau \in \mathbb{R}$ dessen Radius angibt. Dadurch ist die Norm der Vektoren von vornherein fixiert und kann somit nicht uneingeschränkt wachsen.
Mit Hilfe der Maximum-Likelihood-Methode (Kapitel 2.1.3) lautet das Opti-

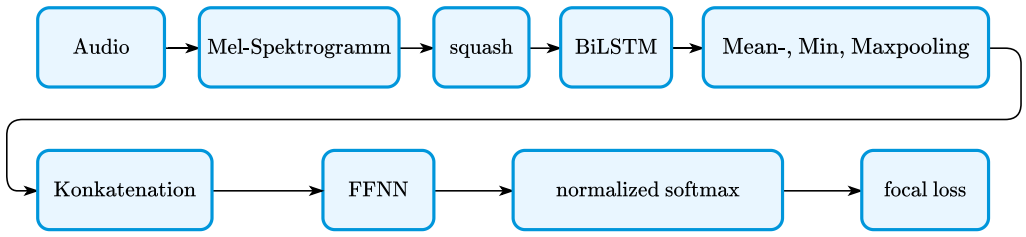


Abbildung 3.1: Bestandteile des vorgeschlagenen Ansatzes.

mierungsziel nun

$$\arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log\left((p(y_i|x_i, \{\theta, W\})\right) \tag{3.10}$$

$$= \arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log\left(\mu(x_i)_{y_i}\right) \tag{3.11}$$

Mit dem Focal Loss [Lin+17] (Gleichung 3.11) wird der Skalierungsfaktor $(1 - \mu(x_i)_{y_i})^{\gamma}$ hinzufügt,

$$\arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} (1 - \mu(x_i)_{y_i})^{\gamma} \log\left(\mu(x_i)_{y_i}\right) \tag{3.12}$$

der die automatische Reduzierung des Einflusses der einfachen Beispiele während des Trainings und die schnelle Fokussierung des Modells auf schwierige Beispiele bewirkt. Je höher γ, desto mehr kommt dieser Effekt zum Tragen. Schwierige Beispiele sind informativer und entscheidend für eine gute Generalisierungs- und Klassifizierungsleistung.
Eine Übersicht über die zentralen Bestandteile des Ansatzes ist in Abbildung 3.1 zu finden.

3.1.3 Datensatz und Experimente zur Evaluation der Architektur

In diesem Abschnitt wird zunächst auf den Datensatz eingegangen, der zur Klassifikation von Primatenvokalisationen verwendet wurde. Darauf folgt die Diskussion der einzelnen Vorverarbeitungs- und Trainingsschritte. Schließlich werden die Experimente vorgestellt, die zur Untersuchung der Wirksamkeit des Ansatzes durchgeführt wurden. Schließlich werden die Ergebnisse interpretiert und eigeordnet.

3.1.3.1 Datensatz mit Vokalisationen von Primaten

Um die Effektivität des Ansatzes zur Klassifizierung von Primatenvokalisationen zu untersuchen, wurde ein kürzlich veröffentlichter Datensatz [Zwe+21]

verwendet. Der Datensatz besteht aus annotierten Aufnahmen von vier verschiedenen Primatenspezies (Schimpansen, Meerkatzen, Mandrillen und Rotkopfmangaben) sowie Hintergrundgeräuschen (vergleichbar mit natürlichen Waldgeräuschen). Das vorliegende Klassifikationsproblem umfasst folglich fünf verschiedene Klassen. Die Aufnahmen wurden in einem Tierschutzgebiet in Kamerun gesammelt und bilden lose die Klassenverteilung ab, die man bei der Überwachung von Primaten in dieser Gegend erhalten würde (6652 Aufnahmen von Schimpansen, 2623 von Mandrillen, 627 von Rotkopfmangaben und 476 von Meerkatzen). Er ist dementsprechend stark unausgeglichen, was in Abbildung 3.4 b) verdeutlicht wird. Darüber hinaus sind in Abbildung 3.2 einige Mel-Spektrogramme der einzelnen Primatenarten aufgeführt.
Die Daten sind in Trainings-, Validierungs- und Testdatensatz aufgeteilt. Während die Trainingsdaten lediglich zur Optimierung verwendet werden, dient der Validierungsdatensatz dazu, die Performanz des Modells objektiv anhand von Daten, auf denen noch nicht trainiert wurde, zu bewerten. Der Validierungsdatensatz wird typischerweise genutzt, um geeignete Hyperparameter zu finden. Auf dem Testdatensatz wird schließlich das Modell mit der besten Hyperparameter-Konfiguration evaluiert, um das Szenario nachzustellen, in dem der Klassifikator im Produktionsbetrieb auf neue, zuvor niemals gesehene Daten trifft.

Hyperparameter	Prior	**Beste Belegung**
Optimierung		
Lernrate lr	$\log\mathcal{U}(10^{-6}, 10^{-1})$	$3.373 * 10^{-4}$
Batch Größe bs	$32, 64, 96 \dots 256$	32
Norm. Softmax nutzen	$\mathbb{B}$	True
Norm. Softmax Temperatur τ	$\mathcal{U}(0.01, 1)$	0.6203
L2 Gewichtsstrafe λ	$\mathcal{U}(10^{-6}, 10^{-1})$	0.0578
Lernratenverfall β	$0.8, 0.82, \dots 1.0$	0.82
Focal loss nutzen	$\mathbb{B}$	True
Focal loss γ	$\mathcal{U}\{1, 5\}$	3
Augmentations		
Frequenzmaskierung p_{fm}	$0, 0.05, \dots 0.5$	0.0
Zeitmaskierung p_{tm}	$0, 0.05, \dots 0.5$	0.0
Anwendungswahrscheinlichkeit p_{r}	$\mathcal{U}(0, 1)$	/
Modell		
LSTM Schichten l	$\mathcal{U}\{1, 4\}$	2
LSTM dim. h	$32, 64, 96 \dots 256$	256
Dropout p_{drop}	$0, 0.05 \dots 0.8$	0.05
# trainierbare Parameter		$\approx$ 3.5Mio.

Tabelle 3.1: Zusammenfassung aller Hyperparameter, ihrer A-Priori Verteilung sowie der besten Belegung.

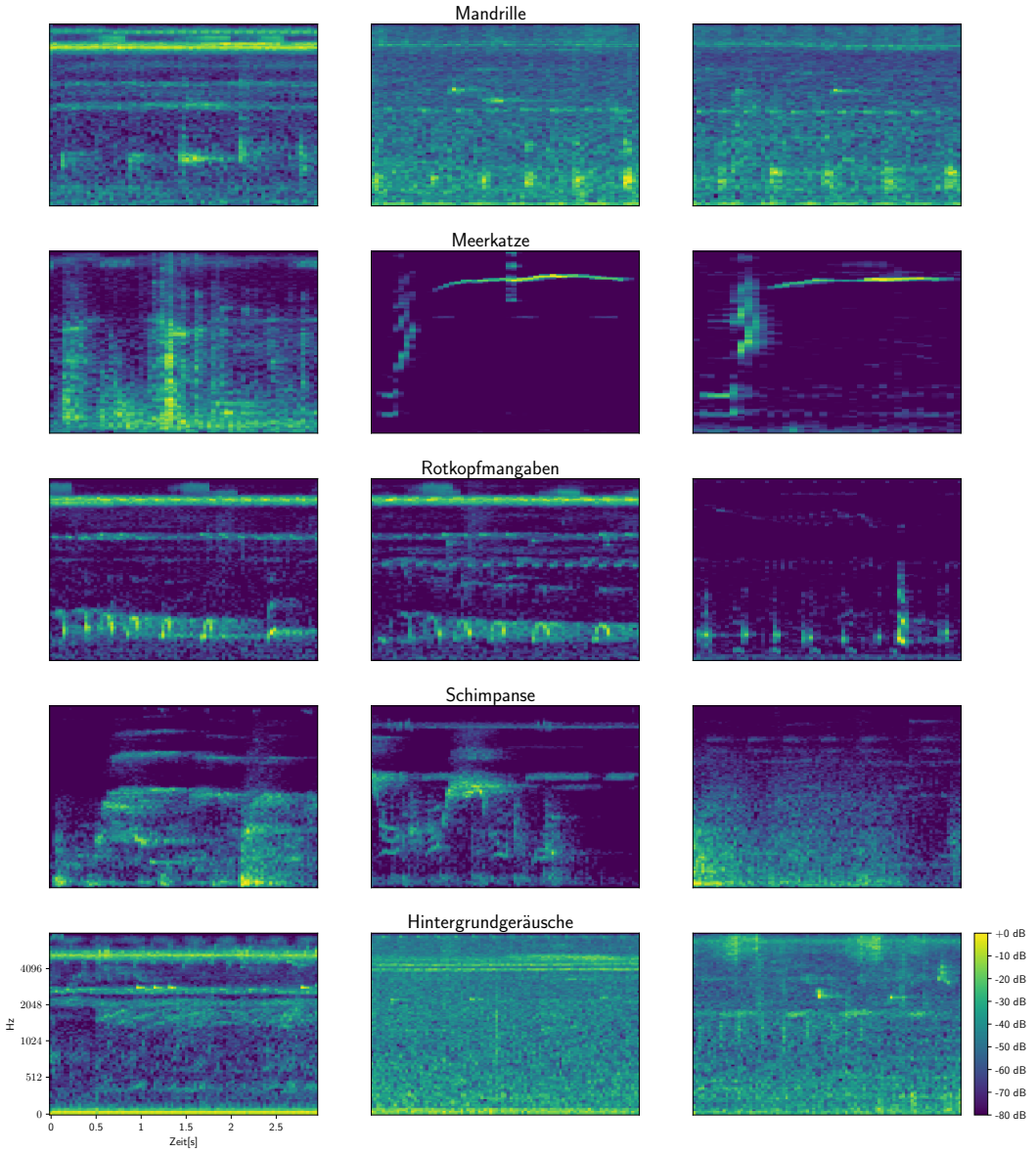


Abbildung 3.2: Exemplarische Mel-Spektrogramme für die einzelnen Primatenarten sowie der Hintergrundgeräusche.

3.1.3.2 Training des rekurrenten neuronalen Netzes

Im Folgenden werden die einzelnen Aspekte des Modelltrainings näher erläutert.

Modell: Zunächst werden die Spektrogramme (128 Mel-Bänder) mit Hilfe von Gleichung 3.3 komprimiert. Die komprimierten Zeitrahmen werden anschließend direkt an ein BiLSTM mit l Schichten und versteckter Dimension h übergeben. Die aggregierte LSTM-Ausgabe wird durch einen FFNN mit drei versteckten Schichten ($3h \rightarrow 2h$, $2h \rightarrow 2h$ und $2h \rightarrow h$) geleitet. Zwischen den einzelnen Schichten werden Dropout mit Wahrscheinlichkeit p_{drop} und die ELU-Aktivierungsfunktion [CUH16] verwendet. Schließlich sagt der Normalized Softmax (Gleichung 3.9) die Klassenzugehörigkeit voraus.

Augmentations: Zum Zweck der Datenerweiterung wird die Frequenzmaskierung auf p_{fm} Prozent der Mel-Bins und die Zeitmaskierung auf p_{tm} Prozent der Zeitfenster angewendet. Maskierung bedeutet hier das Ersetzen eines Zeit- oder Frequenzrahmens durch Nullen. Augmentations werden mit einer Wahrscheinlichkeit von p_{r} angewendet und helfen dabei, die vorliegenden Daten künstlich zu diversifizieren. Dies führt häufig zu geringerer Überanpassung und höherer Klassifizierungsleistung [Par+19; Ill+20].

Optimierung: Dass naive Training tiefer neuronaler Netze auf stark unausgeglichenen Datensätzen führt oft dazu, dass das Netz den einfachsten Weg wählt und nur die häufigsten Klassen vorhersagt. Um dem entgegenzuwirken, wird der Datensatz in jeder Trainingsepoche durch Upsampling der seltenen und Downsamling der häufigen Klassen neu zusammengestellt. Durch dieses Verfahren wird eine gleichmäßige Klassenverteilung erreicht. Das Netzwerk wird mit AdamW [LH18] mit einer Gewichtsstrafe λ, Lernrate lr und einer Batchgröße von bs optimiert. Die Lernrate wird dabei in jeder Epoche mit β multipliziert und das Training für höchstens 300 Epochen durchgeführt.

Hyperparameter Optimierung: Insgesamt umfasst der Ansatz 14 Hyperparameter, für die es eine geeignete Belegung zu finden gilt. Die manuelle Suche nach einer geeigneten Auswahl von Hyperparametern ist zeitaufwendig und kann zu schlechten Ergebnissen führen, da der Prozess oft unstrukturiert ist. Aus diesem Grund wird auf die bayesische Optimierung zurückgegriffen, um einen geeigneten Satz von Hyperparametern zu finden. Zur Optimierung der Hyperparameter auf dem Validierungsdatensatz wird daher ein Tree-Structured Parzen Estimator (TPE) [Ber+11] verwendet. Die Optimierung erfolgt über 400 Versuche, und ein einzelner Versuch wird verworfen, wenn das beste Zwischenergebnis des Versuchs schlechter ist als der Median der Zwischenergebnisse früherer Versuche [Aki+19]. Der Prior für jeden Hyperparameter wurde durch initiale Experimente bestimmt.
Eine Zusammenfassung aller Hyperparameter, ihrer A-Priori Verteilung sowie

der besten Konfiguration ist Tabelle 3.1 zu entnehmen.

	E2Y	DS	OS	OX	AD	Fusion	M1	M2	M_E1-5
Devel.	72.70	81.3	82.4	83.3	84.6	-	<u>89.1</u>	88.9	**89.4**
Test	70.8	78.8	82.2	83.9	86.1	87.5	88.1	<u>88.7</u>	**89.3**

Tabelle 3.2: Vergleich des Ansatzes mit den Baselines. Die Leistung wird anhand des Unweighted Average Recall (UAR) gemessen. Die besten Ergebnisse sind fett gedruckt und die Zweitbesten sind unterstrichen.

3.1.3.3 Leistungsvergleich mit anderen Ansätzen

Um die Qualität des Ansatzes zu bewerten, wird dieser mit den folgenden Baselines aus [Sch+21] verglichen:

i) OpenSMILE (OS) [EWS10]: Eine Menge von 6373 Merkmalen, die aus der Berechnung von Low-Level-Deskriptor-Konturen (LLD) hervorgehen.

ii) Deep Spectrum [Ami+17]: Die Mel-Spektrogramme werden in RGB-Bilder umgewandelt und anschließend durch ein CNN (DenseNet121 [Hua+17]) geleitet. Die Aktivierungen nach der letzten pooling-Schicht werden verwendet, um einen 2048-dimensionalen Merkmalsvektor für jede Aufnahme zu erhalten.

iii) AuDeep (AD) [Fre+17]: In der ersten Stufe werden vier rekurrente Autoencoder unabhängig voneinander auf Mel-Spektrogrammen verschiedener Lautstärkeschwellwerte (-30dB, -45dB, -60dB und -75dB) trainiert. Die Aktivierungen der letzten versteckten Zustände der Autoencoder werden konkateniert, um so einen Merkmalsvektor für jede Aufnahme zu bilden.

iv) OpenXBOW (OX) [SS17]: Für jede Aufnahme werden 65 LLDs und ihre Deltas basierend auf dem COMPARE [Wen+13]-Feature-Set extrahiert und unter Verwendung zweier Codebooks der Größe 1000 quantisiert. Die Histogramme beider Darstellungen werden zur endgültigen Repräsentation konkateniert.

v) End2You (E2Y) [Tzi20]: Drei Schichten eines 1d-CNNs werden genutzt, um Merkmalsvektoren direkt aus dem Rohsignal zu extrahieren. Die Merkmalsvektoren werden anschließend durch Gated Recurrent Units geleitet und die Aufnahmen auf Basis des letzten versteckten Zustands klassifiziert. Es finden also keinerlei Vorverarbeitungsschritte wie die Erstellung von Mel-Spektrogrammen statt.

vi) Fusion [Sch+21]: Hierbei handelt es sich um ein Ensemble aus den fünf oben genannten Ansätzen, welches die Klassenzugehörigkeit durch Mehrheitsentscheidungen (*Majority Vote*) vorhersagt.

Sofern nicht anders angegeben, wird bei den oben vorgestellten Ansätzen eine SVM zur Klassifizierung verwendet.

In Bezug auf das in dieser Arbeit vorgeschlagene Modell werden die folgenden drei Varianten auf dem Trainings- und Validierungsdatensatz evaluiert:

i) M1: Hier wird das beste Modell verwendet, das bei der Hyperparameteroptimierung gemäß Tabelle 3.1 gefunden wurde.

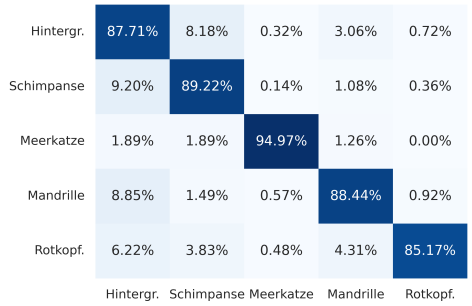
ii) M2: Hier wird das zweitbeste Modell verwendet, das bei der Hyperparameter-Optimierung gemäß Tabelle 3.1 gefunden wurde.

iii) M_E1-5: In Anlehnung an die Fusion-Basislinie werden hier die fünf besten Modelle verwendet, die während der Hyperparameter-Optimierung gefunden wurden. Die finale Klassenzugehörigkeit wird mittels Majority Voting ermittelt.

Aufgrund der Unausgeglichenheit des Datensatzes wird der *Unweighted Average Recall* (UAR) als Evaluationsmetrik verwendet.
Der UAR berechnet sich aus dem Mittelwert des *Recalls* jeder Klasse, wobei der Recall den Quotient aus der Anzahl der korrekt klassifizierten Beispiele einer Klasse und der Gesamtzahl der Beispiele dieser Klasse angibt.
Die Konfusionsmatrizen der Vorhersagen für den Validierungsdatensatz sind in Abbildung 3.3 a) und Abbildung 3.3 b) dargestellt.

Beim Vergleich aller drei Varianten der Modelle in Tabelle 3.2 ist deutlich zu erkennen, dass das Ensemble die beste UAR auf dem Validierungsdatensatz (89.4% vs. 88.9% und 89.1%) und dem Testdatensatz (89.3% vs. 88.1% und 88.7%) erzielt. Dabei wird die Diversität der verschiedenen Modelle im Ensemble durch die Verwendung unterschiedlicher Hyperparameter erreicht. Durch das Majority Voting fallen die Irrtümer eines einzelnen Modells weniger ins Gewicht und können von den anderen Mitgliedern des Ensembles kompensiert werden. Dies wird beim Vergleich der Diagonalen in Abbildung 3.3 a) und Abbildung 3.3 b) deutlich. Das Ensemble verbessert die Klassifizierungsgenauigkeit für alle Klassen außer der Klasse Rotkopfmangabe. Die Einbußen in Bezug auf die Genauigkeit bei dieser Klasse sind darauf zurückzuführen, dass Rotkopfmangaben häufiger für Mandrillen oder einfache Hintergrundgeräusche gehalten werden. Bei sämtlichen Modellen sind Rotkopfmangaben am schwersten zu klassifizieren. Die Ursache hierfür ist das ungünstigere Signal-Rausch-Verhältnis der Aufnahmen von Rotkopfmangaben, da ihre Vokalisationen leiser sind.
Die Vokalisationen von Mandrillen, Meerkatzen und Schimpansen werden am

a) Konfusionsmatrix (bestes Modell)



b) Konfusionsmatrix (Ensemble)

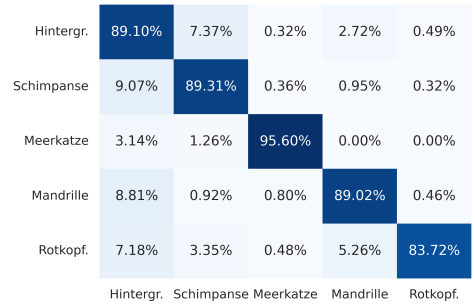


Abbildung 3.3: a) Konfusionsmatrix des besten eigenständigen Modells. b) Konfusionsmatrix des Ensembles.

häufigsten mit Hintergrundgeräuschen verwechselt ($8,81\%$, $3,14\%$ und $9,07\%$). Andererseits werden Hintergrundgeräusche am häufigsten mit Schimpansen verwechselt.
Es ist festzustellen, dass viele dieser Fehler durch Aufnahmen erklärt werden können, bei denen die Vokalisationen schwer hörbar sind, d. h. die Vokalisationen erfolgen aus großer Entfernung zum Aufnahmegerät. Folglich könnten in zukünftigen Arbeiten auch Verfahren zur Reduzierung von Hintergrundgeräuschen eingesetzt werden.
Der vorgestellte Ansatz übertrifft die besten Baselines deutlich. Er erreicht eine UAR von $89,4\%$ im Vergleich zu $84,6\%$ (AuDeep) und $89,3\%$ im Vergleich zu $87,5\%$ (Fusion) auf dem Validierungs- bzw. Testdatensatz. Dies lässt den eindeutigen Schluss zu, dass der Ansatz für die Klassifizierung von Primatenvokalisationen gut geeignet ist.
In Bezug auf die besten Hyperparameter aus Tabelle 3.2 ist zu erkennen, dass der Temperaturparameter τ deutlich größer ist als in vergleichbaren Arbeiten [RCC17; Wan+17; ZW18] empfohlen. Varianten des Normalized Softmax wurden ursprünglich für die Verifizierung von Gesichtern entwickelt, wo die Anzahl der Klassen um mehrere Größenordnungen höher ist, sodass im vorliegenden Fall eine kompaktere Hypersphäre ausreichend ist.
Eine weitere interessante Beobachtung ist, dass die Augmentations vollständig deaktiviert und die Dropoutwahrscheinlichkeit auf einen sehr niedrigen Wert gesetzt wurden ($\tau = 0.05$). Dies deutet darauf hin, dass Augmentations eher Verwirrung stiften als das Lernen zu verbessern. Es ist anzunehmen, dass diese Augmentations besser für Netzarchitekturen wie CNNs geeignet sind, jedoch weitere Untersuchungen erforderlich sind, um diese Hypothese zu bestätigen.

3.1.3.4 Qualtitative Analyse der gelernten Repräsentationen

Die Qualität der gelernten Repräsentationen wird in Abbildung 3.4 a) qualitativ untersucht, indem diese mit TSNE [MH08] visualisiert werden. Jede Klasse erhält dabei eine eigene Farbe. Die Repräsentationen werden anhand

des Validierungsdatensatzes extrahiert.
Bei der Betrachtung der projizierten TSNE-Darstellung in Abbildung 3.4

a) TSNE-Diagram **b)** Klassenverteilung

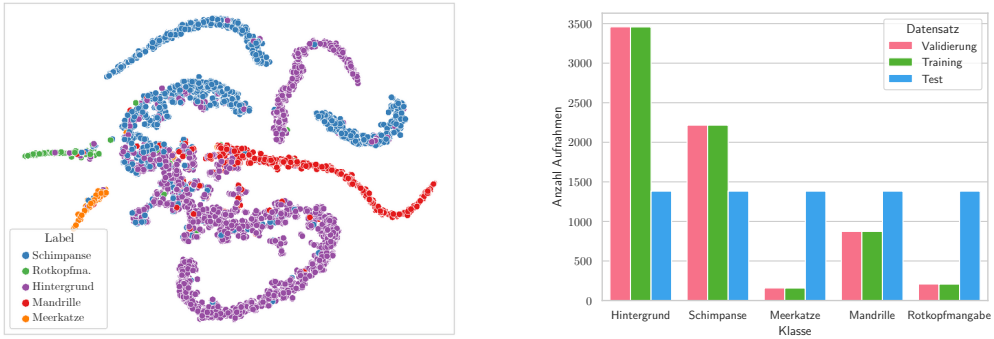


Abbildung 3.4: a) Klassenverteilung in den einzelnen Datensätzen. b) TSNE-Diagramm der Repräsentationen.

a) werden diese Feststellungen erneut bestätigt. Die meisten Punkte sind leicht voneinander zu trennen, jedoch landen gelegentlich Punkte aus anderen Klassen in dem Bereich, in dem die Klasse der Hintergrundgeräusche dominiert.

3.1.4 Fazit und Erweiterungsmöglichkeiten der Architektur

In diesem Kapitel wurde eine rekurrente Netzarchitektur für die Klassifizierung von Primatenvokalisationen vorgestellt. Die Architektur ist inspiriert durch das klassische Feature Engineering, beseitigt aber die Notwendigkeit, manuell einen geeigneten Merkmalssatz zu ermitteln. Darüber hinaus konnte gezeigt werden, dass die vorgeschlagene Architektur alle Baselines übertrifft. Noch bessere Ergebnisse konnten durch die Verwendung des Modells in einem Ensemble erzielt werden.
In zukünftigen Arbeiten könnten zunächst die Auswirkungen des Vortrainings auf einen größeren Datensatz [Fon+21] untersucht werden. So könnte das Netz den Trainingsvorgang bereits mit einer guten Initialisierung starten und so möglicherweise die Klassifikationsergebnisse verbessern. Für das Vortraining besteht zum einen die Möglichkeit, einen annotierten Datensatz zu verwenden und somit ebenfalls ein Klassifikationsproblem zu lösen. Zum anderen wurden in letzter Zeit auch Ansätze entwickelt, die ohne Labels auskommen. Dies kann beispielsweise durch die Vorhersage von aus der Eingabe entfernten Teilen des Mel-Spektrogramms umgesetzt werden [Gon+22]. Eine Methode zum Vortraining, die keine Labels benötigt, könnte dann auch direkt zum Vortraining auf dem vorliegenden Datensatz eingesetzt werden.
Darüber hinaus sind die Kombination des Ansatzes mit CNNs oder die Ver-

wendung von Transformern untersuchenswert.
Schließlich sei angemerkt, dass der vorgestellte Ansatz zwar explizit für die Klassifikation von Primatenvokalisationen entwickelt und optimiert wurde, er aber konzeptionell auch zu anderen akustischen Klassifikationsproblemen passt. Folglich sollte im nächsten Schritt dessen Tauglichkeit im Hinblick auf weitere Anwendungsfelder untersucht werden.

3.2 Akustische Leckerkennung in Wasserversorgungsnetzen

Leckagen sind eine der Hauptursachen für Wasserverluste in Wasserversorgungs- und -verteilungsnetzen. Nicht entdeckte Lecks, die in der Regel auf Korrosion und Bodenbewegungen zurückzuführen sind, können weitreichende negative Auswirkungen auf die umliegende Infrastruktur und Umwelt haben. Überdies entsteht erheblicher finanzieller und ökologischer Schaden. Zu diesem Problem trägt auch die Tatsache bei, dass es unter Umständen sehr lange dauern kann, bis ein Leck entdeckt, lokalisiert und geeignete Gegenmaßnahmen ergriffen werden.
Im Vereinigten Königreich werden täglich etwa 3200 Millionen Liter Wasser durch Leckagen in Wassernetzen verschwendet [Web20], ein unangemessen hoher Wert in Anbetracht des Klimawandels und der Trinkwasserknappheit in vielen Ländern. Es sind folglich größere Anstrengungen erforderlich, um den Wasserverlust durch Leckagen zu reduzieren.
Ein zuverlässiger Indikator für substanziellen Wasserverlust ist die Abweichung vom Durchschnittsverbrauch in einer Bilanzzone bei Nacht. Ist eine Leckage jedoch unauffälliger, wird sie in der Regel erst entdeckt, wenn Wasser aus den Oberflächen austritt und die Bewohner dieses Problem melden.
Die Lokalisierung des Lecks mit Hilfe der akustischen Emission von austretendem Wasser ist eine der am häufigsten verwendeten Lokalisationsverfahren [EZ19] und verläuft wie folgt:
i) Korrelation von Leckgeräuschen: Wenn Flüssigkeit aus einem Rohr austritt, werden Stoßwellen erzeugt, die das Rohr in Schwingungen versetzen, wodurch typische Leckgeräusche entstehen. Mindestens zwei Sensoren werden um die vermutete Leckstelle herum am Leitungsrohr angebracht. Durch die Analyse der unterschiedlichen Ausbreitungszeiten der Leckgeräusche zwischen den Sensoren kann die Stelle weiter eingegrenzt werden. Dieser Ansatz erfordert infrastrukturelle Kenntnisse wie z. B. Rohrmaterial, Durchmesser, Länge und entsprechende Schallgeschwindigkeiten.
ii) Elektroakustische Methode: Zur weiteren Lokalisierung suchen menschliche Leckortungsexperten mit elektronischen Sondierungsstäben nach Leckgeräuschen. Ein Leck kann lokalisiert werden, indem die Tatsache ausgenutzt wird, dass die Leckgeräusche mit der Annäherung des Experten an das Leck intensiver werden. Das Verfahren stützt sich auf qualifizierte Fachkräfte und

erfordert eine zuverlässige Schätzung des Leckstandortes, da die Suche nach den ersten Anzeichen andernfalls sehr zeitaufwändig ausfallen kann.
In dieser Studie soll die elektroakustische Methode mithilfe von maschinellem Lernen automatisiert werden. Mit sieben Kontaktmikrofonen, die an verschiedenen Stellen des Wasserversorgungsnetzes in einem Vorort von München befestigt waren, wurden akustische Aufnahmen des Normalbetriebs sowie von Leckagen erstellt.
Mit diesen Daten werden mehrere unüberwachte Anomalieerkennungsmodelle trainiert und in ein übergeordnetes Verfahren eingebunden, welches mehrere praktische Anforderungen erfüllt, z.B. Energieverbrauch und begrenzte Bandbreiten. Dieser Ansatz kann als erster Schritt in Richtung eines vollautomatischen Leckerkennungssystems verstanden werden, das nicht auf das Auftreten von Leckeffekten angewiesen ist. Darüber hinaus ist der Ansatz datengetrieben und erfordert kein detailliertes mathematisches Modell der Innenstruktur des Wasserversorgungsnetzes.

3.2.1 Bestehende Ansätze zur akustischen Leckerkennung

Verwandte Ansätze der akustischen Leckerkennung basieren meist auf Kreuzkorrelation [MB04; Gao+17], Wavelet-Transformationen [NI02; Tin+19], Support-Vector-Machines [Kan+17; CNT17] oder neuronalen Netzen [Kan+17; CTW19; CTO20]. Andere Methoden kombinieren akustische Daten mit zusätzlichen sensorischen Messungen [Sto+07]. Ihre Leistung hängt vom Material des Rohrnetzes ab (z. B. Polyethylen oder Metall). Keiner dieser Ansätze entspricht jedoch direkt dem Aufbau und den Anforderungen dieser Arbeit, da die Experimente größtenteils unter Laborbedingungen durchgeführt werden und nur einen einzigen algorithmischen Ansatz untersuchen. Andere physikalische Phänomene, die bei Vorhandensein eines Lecks auftreten, können ebenfalls genutzt werden, z. B. durch Thermografie, Bodenradar oder druckbasierte Methoden. Eine ausführliche Zusammenfassung findet sich in [Ade+17; CCZ18].
Neben der Lecksuche hat in letzter Zeit auch das breitere Feld der Erkennung akustischer Anomalien an Bedeutung gewonnen.
Die meisten Arbeiten auf diesem Gebiet basieren auf Autoencodern (AEs). Ein AE lernt, seine Eingabe aus einer komprimierten Darstellung zu rekonstruieren. Es wird angenommen, dass ungesehene, anomale Daten einen höheren Rekonstruktionsfehler aufweisen. Die Ansätze unterscheiden sich vor allem in der verwendeten Architektur.
Duman et al. [DBİ19] verwenden ein Convolutional AE (CAE) auf den Mel-Spektrogrammen von Geräuschen aus industriellen Prozessen. In [MK19] kommen die Autoren ebenfalls zu dem Schluss, dass CAEs zur Erkennung akustischer Anomalien gut geeignet sind, während sie auch die One-Class Support Vector Machine als starken Konkurrenten ausgemacht haben. Koizumi et al. [Koi+17] verwenden ein klassisches FFNN in Verbindung

mit einer neuartigen Fehlerfunktion, die auf statistischen Hypothesentests basiert. Dieser Ansatz erfordert die Simulation anomaler Geräusche durch die Verwendung von Rejection Sampling.
Andere Ansätze untersuchen Methoden, die direkt auf der rohen Wellenform arbeiten [Hay+18; RM19]. WaveNet-ähnliche [Oor+] generative Architekturen, die kausal-dilatierte Faltungen verwenden, werden zur Vorhersage des nächsten Zeitrahmens verwendet. Diese Ansätze sind deutlich parameter- und rechenintensiver als solche, die auf dem Mel-Spektrogramm des Signals arbeiten.

3.2.2 Konkretisierung des Problems der akustischen Leckerkennung

Ein automatisches Leckerkennungssystem sollte energieeffizient, einfach zu betreiben und leicht aktualisierbar sein. Die Problemstellung wird aus dem folgenden Einsatzszenario abgeleitet:
Kleine batteriebetriebene IoT-Geräte nehmen auf Anforderung einer zentralen Steuereinheit Geräusche über Kontaktmikrofone auf. Während eines vordefinierten Zeitraums werden periodisch Geräusche für eine kurze Zeitspanne (2-5s) aufgezeichnet. Anschließend wird die Sammlung kurzer Audiosequenzen über ein energie- und bandbreitenarmes Funknetzwerk [SWM18] an die zentrale Steuereinheit übertragen. Dies reduziert den hohen Energie- und Bandbreitenverbrauch, den eine dauerhafte Übertragung bewirken würde. Die zentrale Steuereinheit kann dann alle erhaltenen Messungen miteinander in Kontext setzen, um zu entscheiden, ob ein Leck vorliegt oder nicht. Darüber hinaus wird durch die Verwendung einer zentralen Steuereinheit die Aktualisierung des Systems erleichtert (z. B. die Anpassung des Algorithmus´ oder das Nachtrainieren des Modells). Diese Vorgehensweise orientiert sich an der Art und Weise, wie menschliche Lecksuch-Experten ihre Einschätzung mithilfe von elektronischen Sondierungsstäben abgeben. Nach dem Anbringen des Stabs an einem Hydrantenanschluss hört der Experte bis zu zehn Sekunden lang konzentriert auf die Quelle. Bei deutlich hörbaren Störgeräuschen, die von der Verkehrsinfrastruktur, der Landwirtschaft oder der Großindustrie emittieren, wird der Vorgang so lange wiederholt, bis eine genaue Einschätzung möglich ist. Zur genauen Leckortung werden verschiedene Hydranten- oder Ventilanschlüsse in der Umgebung entsprechend überprüft. Zunehmende Leckgeräusche deuten auf eine geringere Entfernung zum Leck hin. Die vorliegende Studie behandelt die Leckerkennung.
Aus dem vorgestellten Szenario leitet sich folgende Problemstellung ab:

Problemdefinition 3.1: Leckerkennung in Wassernetzen

Sei X eine Darstellung einer h-minütigen Aufnahme eines an einer Wasserleitung angebrachten Körperschallmikrofons.
Ferner sei $\mathcal{F}_\theta : Y \to \mathbb{R}_+$ eine trainierbare Funktion mit Parametern θ, wobei Y eine Stichprobe von X mit einer Länge von t Sekunden ist.
Wende $\mathcal{F}_\phi$ auf m verschiedene Abschnitte von X an, um einen Vektor $\in \mathbb{R}_+^m$ positiver reellwertiger Anomaliebewertungen zu erhalten.
Um eine einzelne Anomaliebewertung für X zu berechnen, kombiniere die Messungen mit einer Aggregationsfunktion $\phi : \mathbb{R}_+^m \to \mathbb{R}_+$.
Ermittle $(\mathcal{F}_\theta, \phi, h, t, m)$ so, dass die Anomaliebewertungen für Leck-Abschnitte höher sind als die für Nicht-Leck-Abschnitte.

Der zugehörige Algorithmus ist in Algorithmus 1 abgebildet. Es ist zu beachten, dass die Messzeitpunkte gleichmäßig über die gesamte Aufnahme verteilt sind, d.h. linspace $: \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}^m$ liefert den Vektor der Abtastpunkte, vgl. Zeile 2.

$$\text{linspace}(x, y, z) = [\lfloor \frac{(y-x)}{z} \rfloor, \lfloor \frac{2 * (y-x)}{z} \rfloor, \ldots \lfloor \frac{z * (y-x)}{z} \rfloor] \tag{3.13}$$

Jede Messung wird in Zeile 6 vorverarbeitet, um dem Definitionsbereich der Bewertungsfunktion zu entsprechen (z.B. Mel-Spektrogramm, Merkmalsvektor). In Abbildung 3.5 b) sind die wichtigsten Bestandteile des Verfahrens auch visuell dargestellt. Damit genügend Aufnahmen von Leckgeräuschen für das überwachte Lernen zur Verfügung stehen, müsste zunächst eine erhebliche Anzahl von Leckagen künstlich erzeugt werden. Um eine hinreichende Diversität dieser Aufnahmen zu erreichen, müssten zudem verschiedene Abschnitte des Wasserversorgungsnetzes einbezogen werden. Aufgrund der damit verbundenen finanziellen und ökologischen Risiken wird angenommen, dass $\mathcal{F}_\phi$ nur auf Daten des Regelbetriebs trainiert wird.

Algorithmus 1 LeakDetection$(X, \mathcal{F}_\theta, \phi, m, t, \rho)$

1: $h \longleftarrow$ num_cols(X) ▷ Anzahl der Zeitfenster bestimmen
2: $P \longleftarrow$ linspace$(0, h - t, m)$ ▷ Abtaststartpunkte berechnen
3: $S \longleftarrow [\,]$ ▷ Liste für Anomaliebewertungen initialisieren
4: **for** p in P **do**
5: $\quad x \longleftarrow X[:, p : p + t]$ ▷ Ausschnitt aus Mel-Spektrogramm extrahieren
6: $\quad \tilde{x} \longleftarrow \rho(x)$ ▷ Ausschnitt x vorverarbeiten
7: $\quad s \longleftarrow \mathcal{F}_\theta(\tilde{x})$ ▷ Anomaliebewertung berechnen
8: $\quad S$.append(s) ▷ Anomaliebewertungen sammeln
9: **end for**
10: **return** $\phi(S)$ ▷ Anomaliebewertungen aggregieren

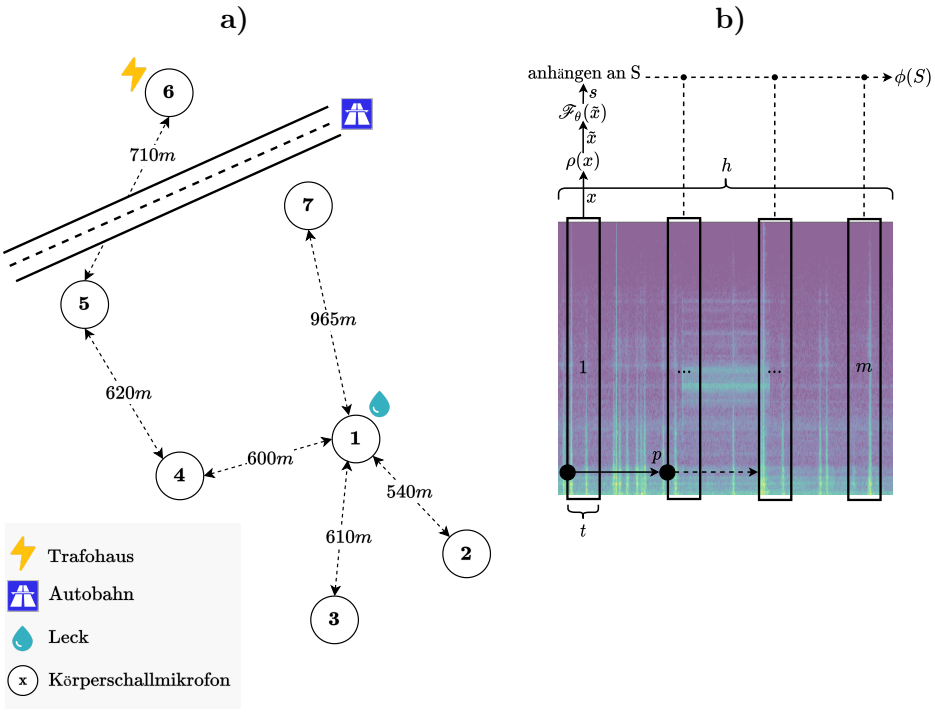


Abbildung 3.5: a) Vereinfachte Karte der Standorte der einzelnen Kontaktmikrofone und signifikanter Objekte in deren Umgebung. Für jedes Kontaktmikrofon ist die Entfernung der kürzesten Verbindung zu einem anderen Kontaktmikrofon angegeben. Die Entfernung ist in Metern der Wasserleitung angegeben.
b) Visualisierung der wichtigsten Aspekte des akustischen Leckerkennungsverfahrens.

3.2.3 Erstellung von akustischen Aufnahmen des Betriebs eines Wasserversorgungsnetzes

Zur Datenerhebung wurde ein verkehrsarmer Vorort von München als Testgebiet ausgewählt. Sieben Körperschallmikrofone wurden direkt an die Hydranten-Verbindungen des Wasserversorgungsnetzes (⌀ 100mm) angebracht. Eine vereinfachte Skizze des Versuchsaufbaus ist in Abbildung 3.5 zu finden.

Die Geräusche wurden über einen Zeitraum von drei Monaten aufgezeichnet. Übermäßig hohe Signale (Übersteuerungen) konnten durch die neutrale Einstellung von Vorverstärker, Equalizer und Nachverstärker vermieden werden. Um verlässliche und vergleichbare Aufnahmen zu erstellen, war es weiterhin essenziel, die dynamische Empfindlichkeit der Körperschallmikrofone zu deaktivieren.

Die resultierenden Aufnahmen haben einen durchschnittlichen Headroom von 24dB. Alle Kontaktmikrofone waren so ausgelegt, dass sie Töne zwischen 300Hz und 3000Hz zuverlässig aufzeichneten, da dies der Frequenzbereich ist, in dem

Leckgeräusche auftreten. Während der Aufzeichnung entstand in unmittelbarer Nähe (ca. 20 m) des ersten Körperschallmikrofons eine mittelgroße Leckage, siehe Abbildung 3.5. Das Leck war 28 Tage lang aktiv und wurde daraufhin geschlossen.
Um die Ausbreitung von Leckgeräuschen zu untersuchen, wurde darüber hinaus folgender Feldversuch durchgeführt. Zunächst wurde durch das Öffnen eines Hydranten in der Nähe von Kontaktmikrofon fünf ein Leck simuliert. Anschließend wurden alle 100 Meter zwischen Körperschallmikrofon vier und fünf weitere Mikrofone angebracht. Ab einer Entfernung von 600 Meter konnte weder das Leck gehört werden, noch waren typische Leckmuster in den Frequenzspektrogrammen sichtbar.

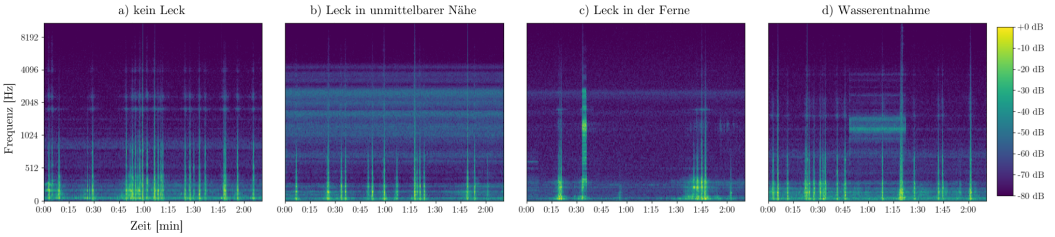


Abbildung 3.6: Vier Mel-Spektrogramme, die a) den Normalbetrieb, b) ein Leck in unmittelbarer Nähe, c) ein synthetisch erzeugtes Leck in der Ferne und d) eine Wasserentnahme veranschaulichen. Die Ausschläge werden durch Autos verursacht, die über die Hydrantenabdeckung oberhalb der Kontaktmikrofone fahren. Andere Aufnahmen können jedoch auch Schritte, Tiergeräusche oder Störgeräusche durch ein nahe gelegenes Trafohaus enthalten. Ein Leck ist durch eine hohe Energie in den oberen Frequenzen gekennzeichnet. Dieses Muster wird mit zunehmender Entfernung zum Leck weniger dominant und verschwindet nach etwa $600m$. Das in b) dargestellte Leck hat einen Durchsatz von ca. $300\frac{\mathrm{ml}}{\mathrm{s}}$.

3.2.4 Ermittlung und Diskussion geeigneter Parameter für den Algorithmus

In diesem Abschnitt wird die Durchführung unterschiedlicher Experimente zur Findung einer geeigneten Belegung von $(\mathcal{F}_\theta, \phi, h, t, m)$ besprochen. Dazu wird zunächst der verwendete Datensatz behandelt, gefolgt von einer kurzen Diskussion der untersuchten Anomalieerkennungsmodelle. Abschließend wird der experimentelle Aufbau besprochen.

Abbildung 3.7: a) Lecksuche mit Hilfe eines Sondierungsstabs. Nachdem der Experte den Sondierungsstab auf die Hydrantenverbindung gesetzt hat, prüft er die Geräusche auf mögliche Anomalien.
b) Hydrantenverbindungen an denen Körperschallmikrofone angebracht wurden.

3.2.4.1 Methodik und Datensatz

Der Untersuchungsaufbau ist so konzipiert, dass er der Problemdefinition (Abschnitt 3.5) entspricht. Hierzu werden 600 Stunden Aufnahmen aus dem Normalbetrieb verwendet, die von allen Körperschallmikrofonen gleichermaßen aufgezeichnet wurden. Die Aufnahmen wurden in Mono mit einer Samplingrate von 16 kHz und einer Bittiefe von 16 Bit erstellt. Ein Bandpassfilter wird angewendet, um Informationen oberhalb und unterhalb der vom Körperschallmikrofon unterstützten Frequenzen zu unterdrücken. Für das Training wurden die Daten in $t = \{2s,\ 5s\}$ lange, überschneidungsfreie Tonproben aufgeteilt. Diese Werte liegen am unteren Ende der Zeitspanne, die ein menschlicher Experte mit einem Sondierstab benötigt, um eine Einschätzung abzugeben. Die Proben werden darauf folgend vorverarbeitet (Algorithmus 1, Zeile 6), indem entweder das Mel-Spektrogramm (Abbildung 3.6) berechnet oder spektrale Merkmale extrahiert werden.
Um die Leistungsfähigkeit des Ansatzes zur Bestimmung von Leck und Nicht-Leck zu bewerten, wurden die folgenden beiden Datensätze erstellt:

i) Leck in unmittelbarer Nähe
Um die Fähigkeit zur Erkennung von Lecks in unmittelbarer Nähe eines Körperschallmikrofons zu messen, wurden fünf aufeinanderfolgende Tage ausgewählt, an denen ein Leck in unmittelbarer Nähe von Körperschallmikrofon 1 auftrat. Für jeden Tag wurden die Aufnahmen von Körperschallmikrofon 1 und einem weiteren Körperschallmikrofon verwendet. Auf diese Weise entstand ein ausgewogener Datensatz mit einer gleichen Anzahl von Leck- und Nicht-Leck-Aufnahmen.

ii) Leck in der Ferne
Mit Hilfe dieses Datensatzes wird untersucht, wie gut ein Leck erkannt werden kann, wenn es sich in größerer Entfernung von einem Körperschallmikrofon befindet. Da nur Aufnahmen eines Lecks in der Nähe eines einzelnen Körperschallmikrofons zur Verfügung standen, wurden normale Aufnahmen mit Leckgeräuschen mit einem hohen Signal-Rausch-Verhältnis von +24dB kombiniert. Dabei steht das Rauschen für die Leckgeräusche. Auf diese Weise entstanden synthetisch erzeugte Aufnahmen, die den Eigenschaften eines Lecks in der Ferne entsprechen (Abschnitt 3.2.3). Die synthetische Erzeugung anomaler Aufnahmen ist ein gängiger Ansatz, wenn anomale Daten schwer zu erlangen sind [DBİ19; Pur+19; Koi+19; Soc+15; Sto+15; Nak+16]. Es wurden 96 Stunden an leckfreien Aufnahmen von den Körperschallmikrofonen 2 und 3 ausgewählt. 48 Stunden dieser Aufnahmen wurden mit zufällig ausgewählten Leckgeräuschen gemischt, die aus einem Zeitraum zwischen $0 - 4$ Uhr morgens stammen.

3.2.4.2 Anomalieerkennungsmodelle

Zur Berechnung einer Anomaliebewertung für einzelne Tonproben (Algorithmus 1, Zeile 7) wurden etablierte Dichteschätzer, Ensemblemodelle und tiefe neuronale Netze eingesetzt. Die Modelle können gemäß des Eingabetyps weiter unterteilt werden:

Mel-Spektrogramm:
Die folgenden Modelle erhalten als Eingabe das Mel-Spektrogramm der Tonprobe:

i) Deep Convolutional Auto Encoder (DCAE)
Ein CNN, das die Eingabe in eine niedrigdimensionale Repräsentation komprimiert und dann die Eingabe aus dieser Repräsentation rekonstruiert.

ii) Adversarial Auto Encoder (AAE) [Mak+15]
Ein DCAE, der so regularisiert ist, dass die gelernten Repräsentationen in etwa der Normalverteilung $\mathcal{N}(0, I)$ folgen.

iii) Adversarial Variational Bayes (AVB) [MNG17]
Ein AAE, welcher anstelle einer einzelnen Repräsentation für eine Tonprobe die Modellparameter einer Normalverteilung (Erwartungswert und Kovarianz) berechnet. Jede Tonprobe ist somit durch eine eigene Normalverteilung charakterisiert.

Um die Annahme der geringen verfügbaren Bandbreite zu berücksichtigen, wurden die Anzahl der Mel-Bänder auf 64 begrenzt.
Für das Training wurden $[4, 16, 32]$ Filter mit 2×2 max-pooling zwischen den einzelnen Schichten und ReLU [Aga18] als Aktivierungsfunktion verwendet. Es wurde für 100 Epochen mit einer Batchgröße von 128, einer L2 Gewichtsstrafe von 10^{-6} trainiert und mittels Adam [KB15] mit einer Lernrate von 0.0001 optimiert. Für die Rekonstruktion wurden die inversen Operationen (Dekonvolution und Up-Sampling) in umgekehrter Reihenfolge angewendet.
Zur Anomaliebewertung wurde der Rekonstruktionsfehler verwendet. Im Fall des AVB wurde dieser über 10 Stichproben gemittelt.

Merkmalsvektor:
Auf Basis des Frequenzspektrums jeder Tonprobe werden die folgenden acht Merkmale berechnet: chromagram, spectral centroid, spectral bandwidth, spectral contrast, spectral roll off, spectral flatness, zero-crossing rate und der root-mean-square value. Dabei handelt es sich um Standardmerkmale aus der Literatur [PS20].

i) Gaussian Mixture Model (GMM)
Ein Dichteschätzer, der die zugrunde liegende Wahrscheinlichkeitsverteilung als Linearkombination von Gaußverteilungen modelliert. Die Parameter werden mit dem Expectation-Maximization Verfahren geschätzt.

Hierbei kommen 16 Gaußverteilungen mit diagonalen Kovarianzmatrizen zum Einsatz.

ii) Bayesian Gaussian Mixture Model (B-GMM)
Im Gegensatz zu einem GMM wird dieses Modell mittels Variational Inference [BKM17] trainiert.

iii) RealNVP [DSB17]
RealNVP nutzt eine Folge von invertierbaren Transformationen, die durch ein neuronales Netz modelliert werden, um die Eingabedaten in eine einfache Ausgangsverteilung zu überführen. Hier werden drei Transformationen der Größe 150 und eine Normalverteilung mit Mittelwert Null und unitärer Kovarianz als Ausgangsverteilung verwendet.

iv) Isolation Forest (IF) [LTZ08]
Der IF partitioniert rekursiv den Repräsentationsraum. Je weniger Aufteilungen nötig sind, um einen Merkmalsvektor zu isolieren, desto höher ist dessen Anomaliebewertung. Es werden 120 Basisschätzer verwendet.

Jeder Merkmalsvektor wird standardisiert, indem der Mittelwert subtrahiert und durch die Standardabweichung der einzelnen Merkmale geteilt wird. GMM, B-GMM und RealNVP berechnen die Anomaliebewertung über die Wahrscheinlichkeitsdichte.

Die Leistung des Modells wird anhand der *Area Under the Receiver Operating Characteristics* (AUC) gemessen, die angibt, wie gut ein Modell über alle Klassifizierungsschwellwerte hinweg zwischen Leck und Nicht-Leck unterscheiden kann. Die AUC ist die Standardmetrik zur Bewertung von Modellen zur Erkennung von Anomalien, da sie einen vollständigen Sensitivitäts-Spezifitätsbericht liefert. Sie wird wie folgt berechnet [Koi+20b]:

$$\mathrm{AUC} = \frac{1}{N_- N_+} \sum_{i=1}^{N_-} \sum_{j=1}^{N_+} I_{[0,\infty)} \left(\mathcal{F}_\theta(x_j^+) - \mathcal{F}_\theta(x_i^-) \right) \tag{3.14}$$

$I_{[0,\infty)}(x)$ ist dann 1, wenn $x > 0$ und andernfalls 0.
Dabei sind $\{x_j^+ \mid 0 \leq j \leq N^+\}$, $\{x_i^+ | 0 \leq i \leq N^-\}$ normale bzw. anomale Testdatenpunkte, die nach ihrer Anomaliebewertung in absteigender Reihenfolge sortiert sind. Die Werte N_- und N_+ stehen jeweils für die Anzahl der normalen bzw. anomalen Testdatenpunkte. Gemäß der obigen Formel werden die Anomaliebewertungen der Testdatenpunkte des Normalfalls als Schwellenwert verwendet. Im Gegensatz zu anderen Klassifikationsmetriken wie z.B. der Genauigkeit muss also kein singulärer Schwellenwert ermittelt werden. Die Anomaliebewertung eines einzelnen anormalen Datenpunktes ist bei der AUC definiert als der Anteil von Normalfalldaten im Testdatensatz, welche eine geringere Anomaliebewertung haben. Die Leistung von $\mathcal{F}_\theta$ wird somit anhand

	Leck in unmittelbarer Nähe				Leck in der Ferne			
h	30min		60min		30min		60min	
t	2s	5s	2s	5s	2s	5s	2s	5s
GMM	74.8±2.5	88.0±1.3	74.6±2.1	87.6±1.6	64.0±5.5	68.7±0.0	68.4±1.0	70.0±1.2
B-GMM	78.7±3.7	88.1±2.4	78.8±4.0	87.7±2.6	64.0±5.1	70.5±1.8	69.3±1.2	72.5±2.9
IF	98.6±0.0	98.8±0.0	**98.8**±0.0	**100**±0.0	41.4±0.0	42.5±1.8	41.0±1.5	42.3±2.3
RealNVP	98.6±2.0	98.0±3.6	99.0±1.3	98.1±3.7	75.2±2.7	**76.1**±2.5	77.1±2.8	**78.3**±3.1
DCAE	98.2±0.0	98.9±0.0	**99.8**±0.0	**100**±0.0	75.1±1.5	73.4±5.0	76.1±2.1	76.0±3.3
AAE	**98.9**±0.0	99.0±0.0	**99.8**±0.0	99.8±0.0	75.0±4.4	69.9±3.2	77.0±5.2	71.6±2.4
AVB	98.1±0.5	**99.5**±0.1	99.6±0.5	**100**±0.0	**76.9**±3.4	75.5±2.7	**78.9**±3.7	77.5±2.8

Tabelle 3.3: Durchschnittliche AUCs ± eine Standardabweichung für verschiedene Belegungen von $\mathcal{F}, h$ und t wobei ϕ = median, $m = 20, h =$ 30min und $m = 40, h =$ 60min.

eines Wertes zwischen 0 und 1 gemessen. Je höher dieser Wert, desto besser kann $\mathcal{F}_\theta$ zwischen Anomalien und dem Normalfall unterscheiden.

3.2.4.3 Modellauswahl

Der wichtigste Aspekt des Ansatzes ist die Wahl einer geeigneten Bewertungsfunktion $\mathcal{F}_\phi$. Die in Abschnitt 3.2.4.1 vorgestellten Datensätze sind in konsekutive h = 30min und h = 60min lange Aufnahmen aufgeteilt. Für jede dieser Aufnahmen wird der Algorithmus 1 separat ausgeführt, um eine einzige Anomaliebwertung zu erhalten. Hier wurde der Median als Aggregationsfunktion ϕ aufgrund seiner Robustheit gegenüber Ausreißern gewählt[1] und über alle Modelle aus Abschnitt 3.2.4.2 mit t = 2 und t = 5 evaluiert. Ferner wird m = 20 und m = 40 für h = 30 min bzw. h = 60 min festgelegt. Die Ergebnisse sind in Tabelle 3.3 dargestellt.

Im Fall von „Leck in unmittelbarer Nähe“ ist die Leistung von GMM und B-GMM im Vergleich zu allen anderen Modellen deutlich geringer. Interessanterweise steigen die AUC-Werte für GMM und B-GMM um etwa 12%, wenn t = 5. Diese Feststellung gilt auch für die andere Konstellation. Bei h = 60min und t = 5s erreichen IF, DCAE und AVB perfekte Werte. Im Allgemeinen kann man diese Aufgabenstellung als trivial für IF, RealNVP, DCAE, AVB und AAE betrachten. Zu beachten ist, dass im Falle von AAE und AVB auch versucht wurde, die logarithmische Wahrscheinlichkeit der Repräsentation zu verwenden, doch erwies sich der Rekonstruktionsfehler als besseres Kriterium. Die Ergebnisse für „Leck in der Ferne“ zeichnen ein anderes Bild. Hier sind die Unterschiede zwischen Mel-Spektrogrammen mit und ohne Leck subtiler und daher wesentlich schwieriger zu erkennen. IF versagt bei dieser Aufgabe vollständig aufgrund des geringeren Abstands zwischen Normalfalldaten und Lecks im Merkmalsraum. Die Anzahl der Raumteilungen ist für fast alle Daten-

[1] Der Median zeigte leicht bessere Ergebnisse im Vergleich zu anderen Aggregationsfunktionen wie dem Mittelwert.

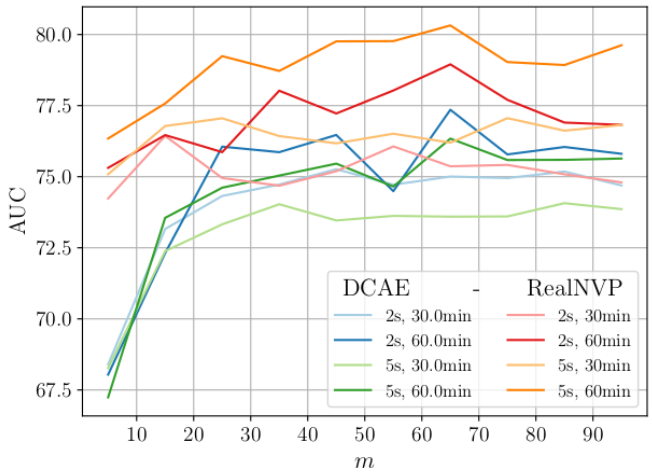


Abbildung 3.8: Leistungsvergleich für Lecks in der Ferne für verschiedene Abtasthäufigkeiten $m = \{5, 15, 25, \ldots 105\}$.

punkte gleich und hängt hauptsächlich von den zufälligen Aufteilungspunkten ab. Überdies ist eine generelle Überlegenheit, der auf neuronalen Netzen (NN) basierenden Methoden RealNVP, DCAE, AVB und AAE zu beobachten, was darauf hindeutet, dass NNs die feineren Unterschiede besser aufdecken.
Bei $t = 2$s schneidet die Mel-Spektrogramm-basierte AVB am besten ab und bei $t = 5$s übertrifft RealNVP alle anderen Methoden. Darüber hinaus lässt sich verifizieren, dass der extrahierte Merkmalsatz tatsächlich ein guter Leckindikator ist, da er in Verbindung mit dem NN-basierten RealNVP sehr gute Resultate erzielt.

3.2.4.4 Anzahl der Stichproben

In diesem Experiment wird der Einfluss der Anzahl der Stichproben auf die Modellleistung untersucht. In Abbildung 3.8 wird die Anzahl der Stichproben m von 5 bis 105 in Schritten von 10 Stichproben variiert. Die Auswertung erfolgt für alle Kombinationen von Entscheidungshorizont $h \in \{30\text{min}, 60\text{min}\}$ und Stichprobenlänge $t \in \{2\text{s}, 5\text{s}\}$ auf „Leck in der Ferne“ mit ϕ=median. Der besseren Übersichtlichkeit halber werden nur die Ergebnisse für RealNVP und DCAE gezeigt.

In Abbildung 3.8 ist zu erkennen, dass die Verwendung von weniger als 15 Stichproben zu einer deutlich schlechteren Leistung führt. Die Ergebnisse stabilisieren sich danach und die optimale Leistung wird bei $m = 65$ für die meisten Konfigurationen erreicht. Daraus lässt sich ableiten, dass sich eine Erhöhung der Stichprobenanzahl positiv auf die Leistung auswirkt, denn die finale Anomaliebewertung hängt dadurch weniger von individuellen Stichproben ab. Durch die Verwendung von mehr Stichproben wird die zugrunde liegende Verteilung besser repräsentiert, da ein Leck durch ein stetiges akustisches Muster

gekennzeichnet ist. Im Gegensatz zu einer konstanten Überwachung der akustischen Emissionen (Streaming-Verfahren) muss jedoch nur ein Bruchteil aller möglichen Messungen berücksichtigt werden, denn es konnte keine wesentliche Leistungssteigerung durch die Berücksichtigung von zusätzlichen Stichproben festgestellt werden.
Die vorgeschlagene Methode erfordert daher lediglich Tonaufnahmen mit einer Gesamtdauer von $65 * 5\text{s} \approx 5.4\text{min}$ ($t = 5\text{s}$) bzw. $65 * 2\text{s} \approx 2.3\text{min}$ ($t = 2\text{s}$).

3.2.5 Fazit und offene Forschungsfragen

In dieser Arbeit wurde ein umfassendes Verfahren zur Erkennung von Lecks in Wassernetzen vorgestellt, das den Anforderungen der Praxis gerecht wird. Weiterhin wurde sorgfältige Evaluierung der verschiedenen erforderlichen Konfigurationsparameter vorgenommen. Während ein Leck in unmittelbarer Nähe eines Körperschallmikrofons für die Mehrzahl der Bewertungsfunktionen trivial ist, lieferten auf neuronalen Netzen basierende Ansätze bessere Resultate in Bezug auf die Erkennung von weit entfernen Lecks. Außerdem wurde gezeigt, dass es nicht notwendig ist, das System ständig zu messen. Es genügt bereits, einen Teil der Aufnahmen während eines vordefinierten Entscheidungshorizonts zu berücksichtigen.
Künftige Arbeiten könnten sich mit verschiedenen Bewertungsfunktionen befassen, z. B. unter Berücksichtigung der sequentiellen Natur der Aufzeichnungen. Weitere Ansatzpunkte, die es wert sind, erforscht zu werden, sind anspruchsvollere Abtastungs- und Aggregationsstrategien. Die Ausweitung des Ansatzes auf die Lecklokalisierung (z. B. durch Trilateration) stellt den nächsten logischen Schritt dar. Ferner wäre es denkbar, separate Modelle zu trainieren, anstatt ein einziges Modell auf die Daten aller Kontaktmikrofone zu trainieren. Alternativ ließe sich das Modell auf die ID der Kontaktmikrofone oder auf Merkmale der Umgebung konditionieren [HB17]. Ein weiterer wichtiger, noch zu erforschender Aspekt ist, wie das Modell aktualisiert werden kann, wenn neue Daten (möglicherweise aus einem anderen Gebiet) eintreffen [Koi+20a].

3.3 Rekurrente Interpolationsnetze zur Erkennung anomaler Geräusche

Die Erkennung von anomalen Geräuschen ist ein wichtiger und bedeutender Anwendungsbereich der künstlichen Intelligenz in der Industrie. Die akustische Überwachung wird oftmals dann eingesetzt, wenn die visuelle Überwachung unpraktisch oder unmöglich ist. Zum Beispiel mag ein kaputter Motor mit gerissenem Zahnriemen von außen zwar nicht beschädigt erscheinen, beim genauen Hinhören lassen sich jedoch Klappergeräusche feststellen, welche auf den Schaden hinweisen.

Generell erfasst ein System zur Erkennung akustischer Anomalien erhebliche Abweichungen vom Normalfall und meldet den Grad der Abweichung, wodurch Reparaturkosten gesenkt und größere Schäden verhindert werden können. Angesichts immer komplexerer Produktionsanlagen, höherer Fertigungskosten und einer zunehmenden Autonomie der Fertigungsanlagen könnte ein zuverlässiges akustisches Anomalieerkennungssystem in naher Zukunft von entscheidender Bedeutung sein.
In diesem Kapitel wird die akustische Anomalieerkennung für Fabrikmaschinen, wie Lüfter, Ventile, Gleitschienen und Pumpen untersucht.
Da die Gewinnung eines möglichst vollständigen Datensatzes anomaler Aufnahmen einen langwierigen, zeit- und kostenintensiven Datenerhebungsprozess erfordert, wird wie auch schon in Kapitel 3.2 davon ausgegangen, dass nur Aufzeichnungen des Normalbetriebs verfügbar sind. Daher ist das Ziel der Arbeit, ein Modell zu entwickeln, welches den Normalbetrieb der Maschinen modelliert und dabei hohe Anomaliebewertungen für Abweichungen von der Norm und niedrige Werte für Geräusche des Normalbetriebs meldet.
Die Mehrzahl der aktuellen Ansätze verwendet dafür Autoencoder (AEs) [Mar+15; Koi+17; Kaw+19; DBİ19; Koi+19; MK19; BDI21]. Im Kontext der akustischen Anomalieerkennung leitet ein AE das Mel-Spektrogramm des Geräusches durch ein neuronales Encoder-Netzwerk, welches eine niedrigdimensionale Repräsentation berechnet. Anschließend wird die Eingabe anhand der Repräsentation mit einem Decoder-Netzwerk rekonstruiert. Dieser Ansatz beruht auf der Annahme, dass Geräusche, die den während des Trainings vorgekommenen ähnlich sind, einen geringeren Rekonstruktionsfehler aufweisen, als Anomalien, die während des Trainings nicht vorkamen.
Kürzlich wurde jedoch nachgewiesen [Sue+20], dass der Rekonstruktionsfehler für die Ränder des Mel-Spektrogramms weiterhin hoch bleibt, da weniger Kontextinformationen, d.h. keine umgebenden Mel-Spektrogramm-Teile zur Verfügung stehen. Wie in Kapitel 3.1.2 und Gleichung 3.3 gezeigt wurde, kann das Mel-Spektrogramm als Sequenz von Spaltenvektoren (Frequenzspektren) angesehen werden. Im Folgenden wird die Bezeichnung *Zeitfenster* für einen einzelnen Spaltenvektor verwendet.
Um das Problem der hohen Rekonstruktionsfehler der Zeitfenster an den Rändern der Mel-Spektrogramme zu lindern, schlagen die Autoren in [Sue+20] vor, fünf Zeitfenster zu extrahieren und das *mittlere Zeitfenster* aus den verbleibenden vier *Kontextzeitfenstern* mit Hilfe eines FFNN vorherzusagen. Dieses Vorgehen wird über das gesamte Mel-Spektrogramm wiederholt.
Die vorliegende Arbeit baut auf diesen Erkenntnissen auf und schlägt mehrere Verbesserungen vor, welche unter dem Namen *DRINK - **D**eep **R**ecurrent **In**interpolation **N**etwor**k*** zusammengefasst sind. Im Gegensatz zu [Sue+20] ermöglicht DRINK eine variable Anzahl von mittleren Zeitfenster und Kontextzeitfenster und basiert anstelle einfacher FFNNs auf rekurrenten neuronalen Netzen. Somit wird die sequentielle Natur der Maschinengeräusche

explizit berücksichtigt.
Die Auswirkungen verschiedener Zeitfenster-Konfigurationen werden anhand von 16 Datensätzen aus realen Fabrikanlagen ausführlich untersucht [Pur+19]. Weiterhin wird DRINK mit anderen Modellen verglichen und es wird gezeigt, dass DRINK mit der richtigen Konfiguration in der Lage ist, diese in 13/16 Fällen zu übertreffen. Die größten Leistungszuwächse können bei nicht-stationären Maschinengeräuschen beobachtet werden. Hier erweist sich die Verwendung eines breiteren Kontexts und mehreren mittleren Zeitfenster in Verbindung mit RNNs zur Erkennung anomaler Geräusche als besonders effektiv.

3.3.1 Bestehende Ansätze zur Erkennung anormaler Geräusche

Wie bereits im vorangegangenen Abschnitt erwähnt, nutzen die meisten Ansätze zu akustischen Anomalieerkennung Autocoder und verwenden den Rekonstruktionsfehler zur Anomaliebewertung.
In [DBİ19] setzten die Autoren erfolgreich CNNs ein, um anomale Geräusche in industriellen Prozessen zu erkennen und zeigten auf, dass CNN-Autoencoder traditionellere Ansätze wie One-Class Support Vector Machines [Sch+99] übertreffen. Dies wurde auch von Meire et al. [MK19] beobachtet. Hier argumentieren die Autoren, dass auf Autoencodern basierende Lösungen zwar bessere Ergebnisse liefern, klassischere Modelle, welche nicht auf neuronalen Netzen basieren, jedoch kürzere Ausführungszeiten erreichen. Die Abwägung zwischen schneller Ausführbarkeit und höherer Genauigkeit muss immer in Abhängigkeit des Einsatzszenarios getroffen werden. In dieser Arbeit wird die These vertreten, dass es zunächst wichtiger ist, verlässliche Modelle zu entwickeln und erst im nächsten Schritt die Ausführungszeiten zu optimieren sind.
In Anlehnung zu [Mar+15] greift DRINK auf LSTMs anstelle von einfachen FFNNs zurück. DRINK rekonstruiert jedoch immer nur den mittleren Teil des Mel-Spektrogramms und vermeidet somit im Gegensatz zu [Mar+15] die konstant hohen Rekonstruktionsfehler der Zeitfenster an den Rändern.
Koizumi et al. [Koi+17] verwenden Variational Autoencoder [KW14] und aufwändiges Rejection Sampling, um anomale Geräusche synthetisch zu erzeugen. Das Optimierungsziel beabsichtigt, unter Verwendung der erzeugten Geräusche, die Wahr-Positiv-Rate unter einer beliebig niedrigen Falsch-Positiv-Bedingung zu erhöhen. DRINK verwendet keinerlei künstlich generierte Anomalien.
In [Kaw+19] wird ein Ensemble von Autoencodern in Verbindung mit mehreren akustischen Front-End-Algorithmen zur Nachhallreduzierung und Rauschunterdrückung eingesetzt. Dies führt zu einer verbesserten Detektionsleistung, da die Eingabedaten dadurch deutlich weniger durch Rauschen verunreinigt sind. Allerdings führt dies eine Vielzahl von zusätzlichen Komponenten ein, die sorgfältig ausgewählt und abgestimmt werden müssen.

Der Group-Masked-Autoencoder [Gir+20] basiert auf dem komplexen Zusammenspiel von neuronalen Dichteschätzern und Autoencoding, um Anomalien in Maschinengeräuschen zu erkennen. Hier wird die gemeinsame Verteilung als bedingte Wahrscheinlichkeitsverteilung über Mel-Spektrogramm-Zeitfenstern faktorisiert. Der Ansatz ist konzeptionell deutlich komplexer und erlaubt keine parametereffiziente Konfiguration einer unterschiedlichen Anzahl von mittleren Zeitfenstern und Kontextzeitfenstern.
Im Gegensatz zu den oben genannten AE-basierten Ansätzen existieren einige konzeptionell anders gelagerte Ansätze, z.B. mittels Matrixfaktorisierung [GKH19] oder der direkten Anwendung von WaveNet [Oor+] ähnlichen Modellen auf den Rohdaten [Hay+18; RM19]. Letzteres ist jedoch rechnerisch anspruchsvoller.

3.3.2 DRINK: Deep Recurrent Interpolation Network

In diesem Abschnitt wird ein neues Verfahren zur Erkennung anomaler Geräusche vorgeschlagen, das einige der Nachteile früherer Ansätze behebt.
Der Ansatz in [Sue+20] basiert auf der Beobachtung, dass die naive Anwendung eines Autoencoders auf das gesamte Mel-Spektrogramm zu hohen Rekonstruktionsfehlern auf den äußeren Zeitfenstern führt, da dem Netzwerk für diese Fenster weniger Kontextinformationen (umgebende Zeitfenster) zur Verfügung stehen. Um dieses Problem zu beheben, wurde das *Interpolation Deep Neural Network* (IDNN) entwickelt. Das IDNN arbeitet mit insgesamt fünf Zeitfenstern. Dabei wird zunächst das mittlere Zeitfenster entfernt. Anschließend werden die verbleibenden Zeitfenster zu einem großen Merkmalsvektor konkateniert und durch ein FFNN geleitet. So wird eine niedrigdimensionale Repräsentation berechnet, aus der anschließend der zuvor entfernte mittlere Zeitfenster rekonstruiert wird.
So kann DRINK als flexiblere Verallgemeinerung des IDNN verstanden werden und behebt die folgenden Schwachstellen des IDNN:

i) Die Verwendung eines Autoencoders zur Rekonstruktion des mittleren Zeitfensters ist überflüssig, da dessen Flaschenhalsstruktur nur nötig ist, wenn ansonsten eine triviale Lösung möglich wäre. Im Falle des Autoencoders ist die triviale Lösung die Identitätsfunktion $f(x) = x$. Diese existiert beim IDNN nicht, da das Rekonstruktionsziel nicht bereits in der Eingabe enthalten ist. DRINK fasst die Problemstellung als Vorhersage- und nicht als Rekonstruktionsproblem auf und ist folglich nicht auf die Flaschenhalsstruktur angewiesen, welche die Modellierungskapazität einschränkt. Es handelt sich um eine selbstüberwachte Vorhersageaufgabe und nicht um eine Rekonstruktionsaufgabe.

ii) Indem beim IDNN die Zeitfenster zu einem großen Merkmalsvektor zusammengefasst werden, wird der sequentiellen Natur der Daten nicht explizit Rechnung getragen. Wenn die Anzahl der Zeitfenster erhöht wird,

steigt zudem die Anzahl der benötigten Netzwerkparameter aufgrund der naiven Verwendung von FFNNs erheblich an. DRINK nutzt bidirektionale LSTMs anstelle von FFNNs und berücksichtigt so die zeitliche Dimension.

iii) Das IDNN erhält genau vier Kontextzeitfenster und rekonstruiert ein einzelnes mittleres Zeitfenster. Im Gegensatz dazu wird DRINK in dieser Arbeit über eine Vielzahl verschiedener Fenstergrößen evaluiert, wobei die Anzahl der benötigten Parameter konstant bleibt.

Die zweite Verbesserung aus ii) ist eine direkte Folge der Verwendung von LSTMs, da diese eine Instanz von rekurrenten neuronalen Netzen sind, bei denen der verborgene Zustand zum Zeitpunkt t wie folgt berechnet wird:

$$h_t = \mathrm{RNN}_\Theta(x_t, h_{t-1}) \tag{3.15}$$

Die Parameter Θ werden über alle Zeitschritte hinweg geteilt und folglich führt die Anwendung eines LSTM mit einer größeren Anzahl von Zeitfenstern nicht zu mehr Netzparametern.

Zur Herleitung des Optimierungsziels seien zunächst

$$\mathrm{ctr}_c : \mathbb{R}^{F\times T} \to \mathbb{R}^{F\times c} \tag{3.16}$$

$$\mathrm{ctxt}_c : \mathbb{R}^{F\times T} \to \mathbb{R}^{F\times (T-c)} \tag{3.17}$$

wobei ctr_c die $c \in \mathbb{N}$ mittleren Zeitfenster des Mel-Spektrogramms $\in \mathbb{R}^{\times T}$ extrahiert und ctxt_c die mittleren Zeitfenster entfernt.
Im Allgemeinen wird davon ausgehen, dass $\frac{T-c}{2} \in \mathbb{N}$. Diese Annahme gewährleistet, dass die Zeitfenster links und rechts von den mittleren Zeitfenstern gleich lang sind. Da nur Aufnahmen des Normalfalls zur Verfügung stehen, besteht der Datensatz D ausschließlich aus gleich großen Mel-Spektrogrammen

$$D = \{x_i \in \mathbb{R}^{F\times T} \mid 1 \le i \le n\} \tag{3.18}$$

Das Ziel von DRINK ist nun die Maximierung der bedingten Wahrscheinlichkeit der mittleren Zeitfenster gegeben der Kontextzeitfenster:

$$\arg\max_\theta \frac{1}{n} \sum_{i=1}^{n} \log\left((p(\mathrm{ctr}_c(x_i) \mid \mathrm{ctxt}_c(x_i), \theta)\right) \tag{3.19}$$

Unter der Annahme, dass die einzelnen Werte des Kontextzeitfensters normalverteilt mit Einheitskovarianz sind, und ein many-to-one $\mathrm{BiLSTM}_\theta : \mathbb{R}^{F\times(T-c)} \to \mathbb{R}^{F\times c}$ zur Vorhersage der mittleren Zeitfenster verwendet wird,

folgt dann

$$\arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log\left(\mathcal{N}(\mathrm{ctr}_c(x_i) \mid \mathrm{BiLSTM}_\theta(\mathrm{ctxt}_c(x_i)), I)\right) \tag{3.20}$$

Mit der Definition der multivariaten Normalverteilung [Bis06]:

$$\mathcal{N}(\mathbf{x} \mid \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d * \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu)\right) \tag{3.21}$$

Wird $\Sigma = I$ auf die Einheitskovarianz gesetzt, so ist:

$$\mathcal{N}(\mathbf{x} \mid \mu, I) = \frac{1}{\sqrt{(2\pi)^d}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^2\right) \tag{3.22}$$

$\frac{1}{\sqrt{(2\pi)^d}}$ und $\frac{1}{2}$ sind Konstanten und damit für die Optimierung unerheblich. Das konkrete Optimierungsziel lautet also:

$$\arg\max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log\left(\exp\left(-\left(\mathrm{ctr}_c(x_i) - \mathrm{BiLSTM}_\theta(\mathrm{ctxt}_c(x_i))\right)^2\right)\right) \tag{3.23}$$

$$= \arg\max_{\theta} -\frac{1}{n} \sum_{i=1}^{n} \left(\mathrm{ctr}_c(x_i) - \mathrm{BiLSTM}_\theta\left(\mathrm{ctxt}_c(x_i)\right)\right)^2 \tag{3.24}$$

$$= \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left(\mathrm{ctr}_c(x_i) - \mathrm{BiLSTM}_\theta\left(\mathrm{ctxt}_c(x_i)\right)\right)^2 \tag{3.25}$$

DRINK minimiert folglich die quadratische Differenz (*Least Squares*) zwischen der Vorhersage des BiLSTMs und den tatsächlichen mittleren Zeitfenstern.
Das Verfahren benötigt keine externen Labels, da nur Informationen vorhergesagt werden, die absichtlich aus der Eingabe entfernt wurden. In diesem Sinne ähnelt DRINK neueren selbstüberwachten Ansätzen [Liu+22] aus der Computerlinguistik, die auf der maskierten Sprachmodellierung basieren [Pet+18a; Dev+19]. Bei der maskierten Sprachmodellierung werden Wörter aus Sätzen entfernt und diese dann wieder vorhergesagt. DRINK arbeitet jedoch mit kontinuierlichen und nicht mit diskreten Werten. Eine grafische Darstellung von DRINK ist in Abbildung 3.9 zu finden. Anstatt lediglich die mittleren Zeitfenster zu entfernen, können diese auch durch Nullen $0^{F \times c}$ oder durch eine lernbare Matrix $m \in \mathbb{R}^{F \times c}$ ersetzt werden. In letzterem Fall wird M zu den trainierbaren Parametern des Modells hinzugefügt, d.h. $\theta \leftarrow \theta \cup m$. Damit lassen sich die Eingabedaten gleichförmiger gestalten und das Netz wird explizit über die Position der fehlenden Zeitfenster informiert.

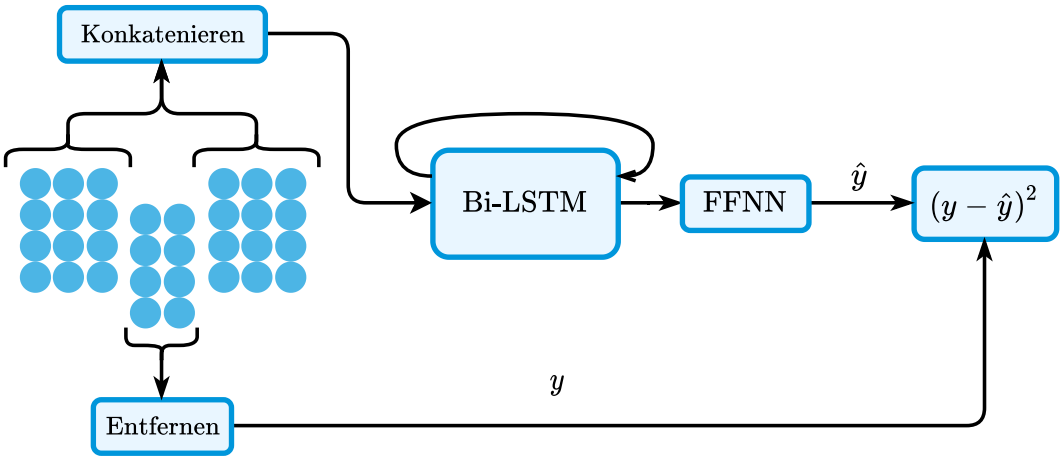


Abbildung 3.9: Visuelle Darstellung der Funktionsweise von DRINK. Zunächst werden die c mittleren Zeitfenster des Mel-Spektrogramms entfernt. Die Kontextzeitfenster werden konkateniert und durch ein bidirektionales LSTM (BiLSTM) geleitet. Der letzte versteckte Zustand des BiLSTM wird anschließend durch einen FFNN geleitet, um eine Vorhersage für die mittleren Zeitfenster zu erhalten. Der Vorhersagefehler wird für das Training und die Erkennung von Anomalien verwendet.

3.3.3 Experimenteller Aufbau und Bewertung der Leistung von DRINK

In diesem Abschnitt soll die Eignung von DRINK für die Erkennung anormaler Geräusche empirisch evaluiert werden. Hierzu wird zunächst der Datensatz vorgestellt, der für die Experimente herangezogen wurde. Im Anschluss daran wird DRINK mit vergleichbaren Verfahren gegenübergestellt und der Einfluss der Anzahl der Zeitrahmen untersucht.

3.3.3.1 Akustische Aufnahmen von Fabrikmaschinen für das Modelltraining

Um die Wirksamkeit dieses Ansatzes zu untersuchen, wird der kürzlich veröffentlichte MIMII-Datensatz [Pur+19] verwendet, der Aufnahmen von Lüftern, Pumpen, Gleitschienen und Ventilen im Normalbetrieb und bei Störungen wie Leckagen, Verstopfungen, Spannungsänderungen, einem losen Riemen, Schienenschäden oder fehlendem Fett enthält. Für jeden Maschinentyp sind Geräusche von vier verschiedenen Maschinenmodellen vorhanden (ID $0, 2, 4$ und 6). Zudem existieren drei verschiedene Versionen jeder Aufnahme, bei denen reale Hintergrundgeräusche aus einer Industrieanlage mit dem Maschinengeräusch vermischt wurden. Wie stark hörbar die Hintergrundgeräusche sind, ist durch den *Signal-to-Noise-Ratio* (SNR) angegeben (6dB, 0dB und -6dB). Der Datensatz enthält insgesamt 26092 Aufnahmen des Normalfalls und 6065

anomale Aufnahmen, die auf $4 * 4 * 3 = 48$ (Maschinentypen * Maschinennummern * SNRs) verschiedene Datensätze verteilt sind. Im Rahmen dieser Arbeit werden die Aufnahmen mit einem SNR von 0dB verwendet, was zu insgesamt 16 verschiedenen Datensätzen führt. Alle Aufnahmen sind mit 16kHz abgetastet und haben eine Laufzeit von 10 Sekunden. Weiterhin wird jede 10s lange Aufnahme in ein Mel-Spektrogramm der Größe 128×313 transformiert. Abbildung 3.10 stellt exemplarisch einige Mel-Spektrogramme der verschiedenen Maschinentypen unter normalen und anomalen Bedingungen dar.
DRINK ermittelt eine Anomaliebewertung für alle mittleren Zeitfenster des Mel-Spektrogramms und berechnet anschließend den Mittelwert der einzelnen Rekonstruktionsfehler. Stehen nicht genügend Kontextzeitfenster zur Verfügung, werden diese wiederholt bis die gewünschte Anzahl erreicht ist. Schließlich werden alle Mel-Spektrogramme Min-Max normalisiert, sodass deren Werte im Bereich $[0, 1]$ liegen.

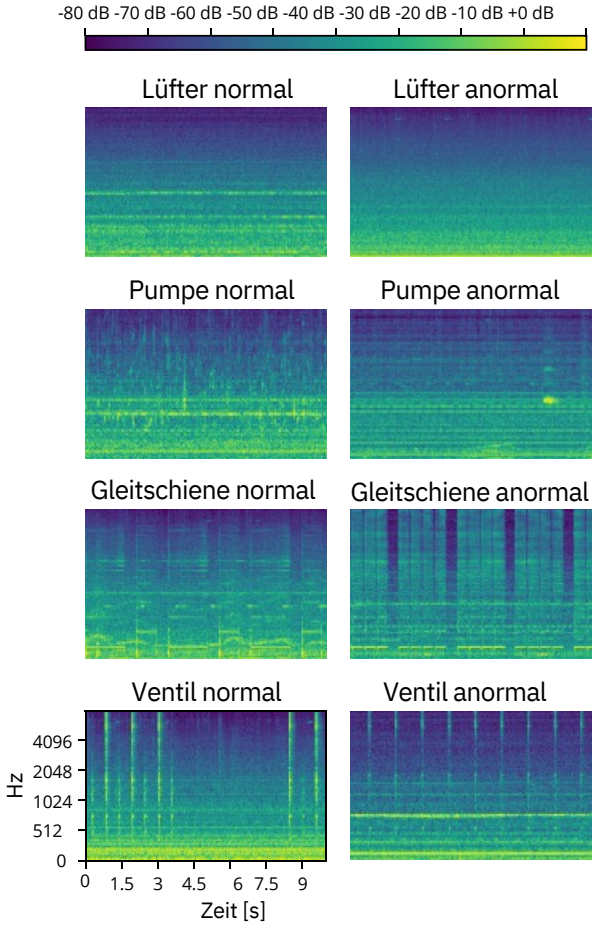


Abbildung 3.10: Exemplarische Mel-Spektrogramme aus dem MIMII-Datensatz [Pur+19] bei normalem und anormalem Betrieb. Gleitschienen und Ventile besitzen ein nicht-stationäreres Klangprofil.

3.3.3.2 Einfluss der Anzahl der Zeitfenster

Einer der größten Vorteile des Ansatzes ist die Unterstützung einer variablen Anzahl von mittleren und Kontextzeitfenstern. Nun wird deren Einfluss auf die Leistung bei der Erkennung anormaler Geräusche genauer untersucht.
In Abbildung 3.11 ist eine Zeile für jeden Maschinentyp und eine Spalte für jede Maschinennummer angegeben. In jeder Abbildung ist auf der x-Achse die Anzahl der vorhergesagten mittleren Zeitfenster $c \in \{1, 2, 4, 6\}$ angegeben. Die y-Achse entspricht der Leistung, die anhand AUC ermittelt wurde und die Fehlerbalken sind in Form von $\pm$ einer Standardabweichung angegeben. Weiterhin wird für jedes c zusätzlich die Anzahl der Zeitrahmen $T \in \{5, 6, 10, 11, 18, 19, 26, 27, 35, 35, 42, 43, 51, 52, 58\}$ variiert und die verschiedenen Konfigurationen farblich verdeutlicht. Zur besseren Verständlichkeit wird in Abbildung 3.11 eine Kleidergrößenanalogie (XXS - XXXL) für T verwendet.
Es gilt zu beachten, dass ungerade Zahlen für T nur für $c = 1$ verwendet werden, um eine gleiche Anzahl von Zeitfenstern links und rechts des mittleren Fensters zu gewährleisten. Für $c = 6$ und $c = 4$ wird der Wert $T = 6$ ausgelassen, da dies zu einer sehr geringen Anzahl von Kontextzeitfenstern (0 bzw. 2) führen würde. In diesen beiden Fällen wird zusätzlich eine weitere Konfiguration mit $c = 58$ berechnet. Die gestrichelte Linie in jeder der 16 Konfigurationen kennzeichnet die mittlere Leistung der IDNN-Baseline aus Abschnitt 3.3.3.3.

Bei Lüfter- und Pumpengeräuschen ist zu beobachten, dass die Leistung mit zunehmender Anzahl an Zeitfenstern abnimmt, während die Standardabweichung ungefähr gleich bleibt. Dies deutet darauf hin, dass für die Erkennung von Anomalien bei stationären Geräuschen nur eine geringe Anzahl erforderlich ist und das Hinzufügen weiterer Kontextzeitfenster keinen Mehrwert liefert. Vielmehr wird dadurch die Vorhersageaufgabe erschwert, was zu einer Verschlechterung der Leistung führt. In Bezug auf die Anzahl der mittleren Zeitfenster zeigt sich ein leichter Vorteil für eine höhere Anzahl, mit Ausnahme des Maschinenidentifikators 2. Interessanterweise sind auf Lüfter- und Pumpendatensätzen in 5/8 Fällen die Konfigurationen mit $T = 10$ und $c = 6$, d.h. $T - c = 4$ Kontextzeifenster und 6 mittlere Zeitfenster, die besten Hyperparameter. In diesem Falle ist der Kontext, auf den sich DRINK stützt, kleiner als die Anzahl der mittleren Zeitfenster, die vorherzusagen sind.
Bei Klängen mit einem nicht-stationären Profil, d. h. bei Gleitschienen und Ventilen, ergibt sich ein anderes Bild. Mit Ausnahme von Ventil-2 ist zu erkennen, dass eine Erhöhung von T und damit eine Erhöhung der Anzahl der verfügbaren Kontextinformationen einen positiven Effekt auf die Leistungsfähigkeit hat. Dieser Aspekt ist bei Ventilgeräuschen stärker ausgeprägt als bei Gleitschienen. Bezüglich der Anzahl der mittleren Zeitfenster c im Bezug auf nicht-stationäre Geräusche kann festgestellt werden, dass die Vorhersage von mehreren Zeitfenstern die Erkennungsleistung von anomalen Geräuschen verbessert.

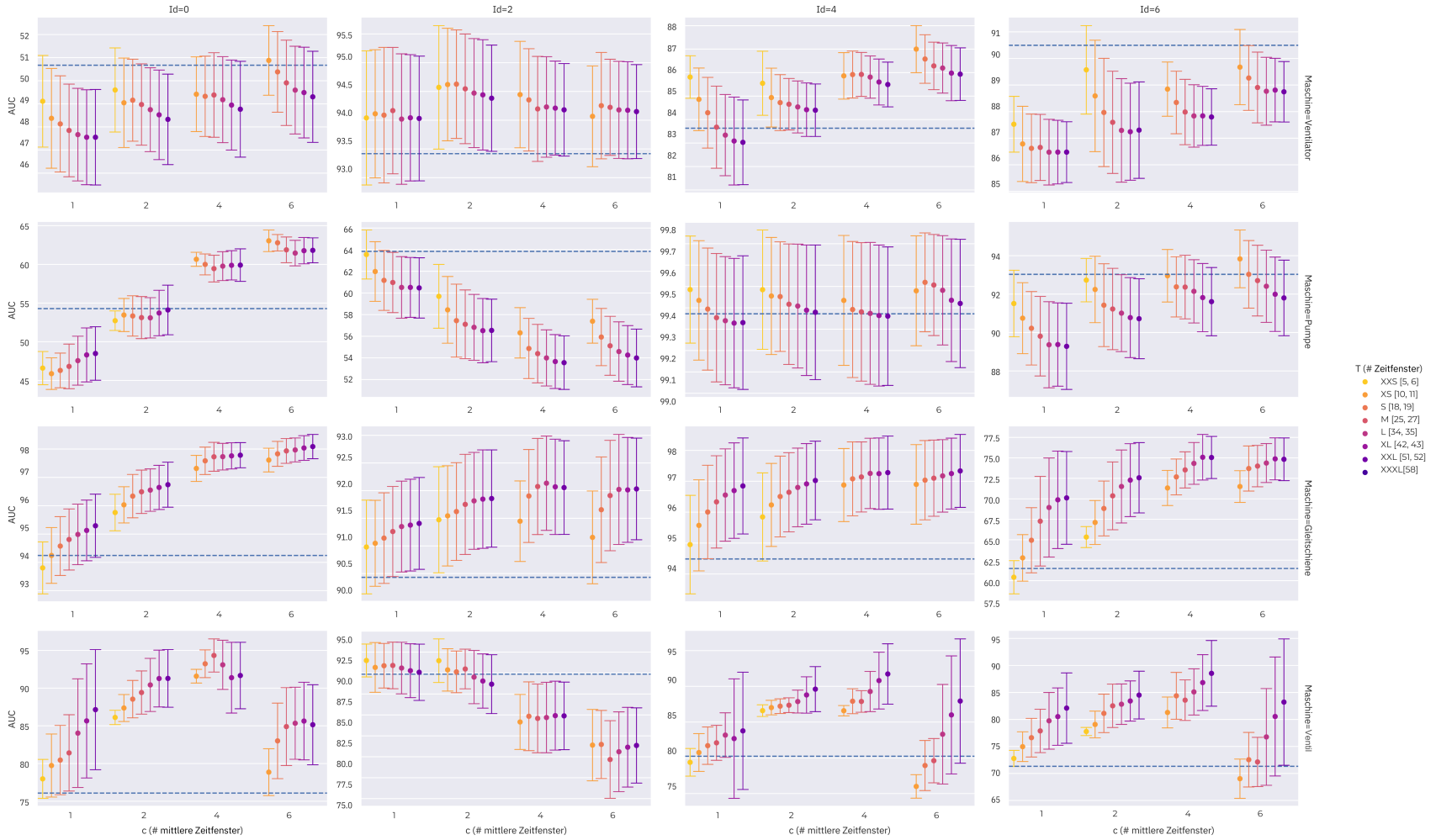


Abbildung 3.11: Untersuchung des Einflusses von DRINK's entscheidenden Hyperparametern c - die Anzahl der mittleren Zeitfenster die DRINK vorhersagen muss und T - die Anzahl der gesamten Zeitfenster. Folglich ist $T - c$ die Anzahl der Kontextzeitfenster, die DRINK als Eingabe erhält. Die gestrichelte Linie kennzeichnet die IDNN-Baseline.

3.3.3.3 Leistungsvergleich mit ähnlichen Ansätzen

Um DRINK's Leistungsfähigkeit bewerten zu können, werden die folgenden Ansätze miteinander verglichen:

i) FFNN-Autoencoder (AE) [Koi+20b]: Ein Autoencoder, der fünf aufeinanderfolgende Mel-Spektrogramm-Zeitfenster rekonstruiert, wie in [Koi+20a] vorgeschlagen. Die Architektur wurde geringfügig angepasst, um der höheren Anzahl von Mel-Bändern Rechnung zu tragen, die in dieser Arbeit verwendet werden.
Der AE besitzt insgesamt 5 versteckte Schichten der Größen $256, 64, 32, 64$ und 256.

ii) IDNN [Sue+20]: Auch hier wurde die Architektur des Interpolation Deep Neural Networks leicht verändert, um der erhöhten Anzahl von Mel-Bändern zu entsprechen. Wie in [Sue+20] nimmt das IDNN vier Zeitfenster als Eingabe und rekonstruiert das mittlere Zeitfenster. Das IDNN umfasst insgesamt 5 versteckte Schichten der Größen $256, 64, 32, 64$ und 256.

iii) DRINK[c, T]: Das Deep Recurrent Interpolation Network erhält $T - c$ Zeitfenster als Eingabe und gibt die c mittleren Zeitrahmen zurück. Hierbei wird ein bidirektionales LSTM mit einem 64-dimensionalen versteckten Zustand genutzt. Die Konkatenation der beiden letzten versteckten Zustände (einer für jede Richtung) wird zur Vorhersage durch ein zweischichtiges FFNN der Größen 128 und 128 geleitet.

Alle Modelle werden bis zur Konvergenz unter Verwendung des ADAM-Optimierers [KB15] mit einer Lernrate von $0,0001$, einer L2-Gewichtsstrafe von $\lambda = 10^{-6}$ trainiert und verwenden die Rectified Linear Unit (ReLU) [NH10] Aktivierungsfunktion.
Tabelle 3.4 fasst die Ergebnisse für alle 16 Datensätze zusammen. Dabei ist für DRINK immer die beste Belegung (siehe Abschnitt 3.3.3.2) von T und c gemäß Abbildung 3.11 angegeben. Als Evaluationsmetrik kommt die AUC (Gleichung 3.14) zum Einsatz. Um die Testdatensätze zu bilden, wird die gleiche Menge an Normalfalldaten zufällig entfernt, wie anomale Daten vorhanden sind. D.h., das Training wird mit den verbleibenden Normalfalldaten durchgeführt, anomale Daten werden während des Trainings nie gesehen und die Testdatensätze sind ausgeglichen. Jedes Experiment wird fünfmal wiederholt.

Zunächst lässt sich in Abbildung 3.11 feststellen, dass es nur zwei Konfigurationen gibt (Lüfter-6 und Pumpe-2), in denen die IDNN-Baseline nicht durch eine der Konfigurationen von DRINK$[T, c]$ übertroffen wird. Es gibt allerdings noch drei weitere Konfigurationen, bei denen die Baseline nur

Maschine	*ID*	AE	IDNN	DRINK	T, c
Lüfter	0	48.01±2.1	50.67±1.7	**50.88±1.7**	10, 6
	2	92.25±2.0	93.42±1.3	**94.62±1.0**	18, 2
	4	79.43±1.5	83.63±1.0	**87.00±1.1**	10, 6
	6	**91.24±1.5**	90.50±1.5	89.67±1.6	10, 6
Pumpe	0	62.14±2.6	54.28±1.2	**63.10±1.6**	10,6
	2	**66.43±3.0**	63.84±3.1	63.53±2.5	5, 1
	4	**99.60±0.4**	99.38±0.3	99.51±0.2	18, 6
	6	92.50±1.9	93.03±1.9	**93.83±1.7**	10, 6
Gleitschiene	0	95.80±1.1	94.24±1.0	**98.14±0.4**	58, 6
	2	90.03±1.2	90.43±1.4	**92.13±1.0**	34, 6
	4	95.78±1.7	94.49±1.9	**97.33±1.2**	50, 6
	6	60.55±2.4	61.48±3.0	**75.10±2.8**	42, 4
Ventil	0	67.10±2.7	76.08±3.5	**94.33±2.3**	26, 4
	2	79.50±3.8	89.86±2.7	**91.50±2.2**	5, 1
	4	70.60±1.5	80.22±2.5	**92.47±4.4**	58, 4
	6	64.1±2.2	71.32±2.4	**89.70±6.4**	58, 4

Tabelle 3.4: Vergleich von DRINK mit anderen Ansätzen. Jede Zelle repräsentiert die durchschnittliche AUC ± eine Standardabweichung.

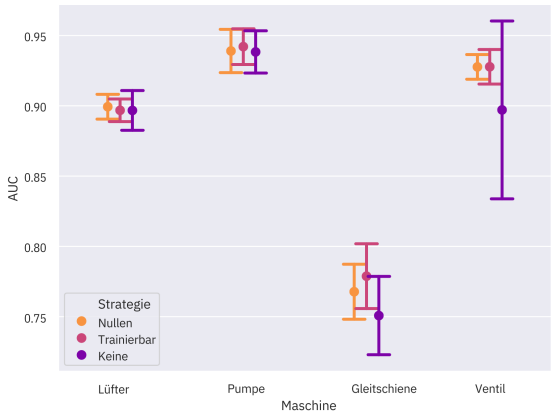


Abbildung 3.12: Leistungsvergleich verschiedener Strategien zum Ersetzen der mittleren Zeitfenster.

geringfügig schnlechter abschneidet (Lüfter-0, Pumpe-4 und Ventil-6). Bei den übrigen Szenarien übertrifft die Mehrheit der $[T, c]$-Belegungen von DRINK das IDNN. Die größten Leistungssteigerungen sind bei Lüfter- und Ventilgeräuschen zu beobachten. Wie bereits in früheren Arbeiten festgestellt [Sue+20; Mül+21b; Mül+21a], sind die von Gleitschienen und Ventilen ausgehenden Geräusche nicht stationär, d. h. sie weisen ein vielfältigeres und sich zeitlich veränderndes Klangmuster auf. Hier ist die Verwendung neuronaler Netze, die die zeitliche Struktur explizit berücksichtigen, von Vorteil.

3.3.3.4 Einfluss der Strategie zum Ersetzen der mittleren Zeitrahmen

Wie am Ende des Abschnitts 3.3.2 kurz erörtert, können entweder einfach die mittleren Zeitrahmen direkt entfernt, durch Nullen oder eine trainierbare Matrix ersetzt werden.
Um die Auswirkung einer jeden Strategie auf die Leistungsfähigkeit zu untersuchen, wird die Maschine ID= 6 mit der besten Konfiguration von T, c aus Tabelle 3.4 ausgewählt. Die Ergebnisse für jede Strategie sind in Abbildung 3.12 angegeben. Aufgrund des hohen Rechenaufwands der einzelnen Experimente wurden nicht alle möglichen Kombinationen berücksichtigt.

Hinsichtlich Abbildung 3.12 ist ein leichter Vorteil für das Ersetzen durch eine trainierbare Matrix oder durch Nullen bei Gleitschienen und Ventilen zu beobachten, nicht aber bei Lüftern und Pumpen. Das Ersetzen der entfernten Zeitfenster führt in jedem Fall zu einer Reduktion der Varianz, was besonders bei Ventilgeräuschen deutlich wird. Dies könnte darauf zurückzuführen sein, dass die Mel-Spektrogramme so explizit Informationen über den Anfang und das Ende des entfernten Ausschnitts enthalten, was zu weniger Verwirrung

beim Training führen kann. Es sind jedoch weitere Experimente erforderlich, um diese Behauptung zu beweisen.

3.3.3.5 Die Grenzen von DRINK

Zu beachten ist, dass DRINK immer noch auf der allgemeinen Annahme beruht, dass anomale Geräusche schwieriger zu rekonstruieren sind und dass Normalfalldaten einen geringen Rekonstruktionsfehler aufweisen.
DRINK verbessert die Detektionsleistung, weil niedrigere Rekonstruktionsfehler für Aufnahmen ermöglicht werden, die dennoch schwer zu rekonstruieren sind, z.B. aufgrund ihrer Nichtstationarität. Schwer vorherzusagende Normalfalldaten würden sonst fälschlicherweise für Anomalien gehalten und die Erkennung von Anomalien erschwert werden.
Abbildung 3.11 zu entnehmen ist, dass während $c = 4$ der beste Wert für Ventilgeräusche ist, beeinträchtigt $c = 6$ die Leistung erheblich. Demzufolge ist eine unvorsichtige Erhöhung von c und T bei nicht-stationären Geräuschen eine schlechte Strategie und stellt DRINK's größten Nachteil dar. Die Hyperparameter T und c müssen sorgfältig gewählt werden, was angesichts knapper Validierungsdaten problematisch sein kann.

3.3.4 Fazit und mögliche konzeptionelle Verbesserungen

In dieser Arbeit wurde DRINK vorgestellt. DRINK sagt auf Basis von Kontextzeitfenstern des Mel-Spektrogramms die mittleren Zeitfenster vorher. Der Vorhersagefehler kann wiederum zur Anomaliebewertung genutzt werden.
Es konnte gezeigt werden, dass DRINK vergleichbare Ansätze übertrifft, wenn die Anzahl der mittleren Zeitfenster und der Kontextzeitfenster richtig gewählt werden.
Zwar konnte nur eine geringe Verbesserung bei Lüfter- und Pumpengeräuschen beobachtet werden, bei nicht-stationären Geräuschen, die von Gleitschienen und Ventilen ausgehen, wurden jedoch große Leistungszuwächse erzielt. Daraus lässt sich ableiten, dass DRINK besonders gut für Szenarien geeignet ist, in denen eine gute Vorhersage von mittleren Zeitfenstern stark von einer Vielzahl von umliegenden Kontextzeitfenster abhängt, was bei komplexeren, nichtstationären akustischen Signalen der Fall ist.
Zukünftige Arbeiten könnten versuchen, ein einzelnes Modell mit unterschiedlichen Konfigurationen zu trainieren und den Vorhersagefehler zu mitteln. Auf diese Weise könnten die Ergebnisse robuster werden und das Problem, die richtigen Hyperparameter zu finden, würde entschärft. Eine zweite Möglichkeit, die Abhängigkeit von den richtigen Hyperparametern zu verringern, stellt die Verwendung eines großen Ensembles über eine Vielzahl von unterschiedlichen Konfigurationen dar.

3.4 Kapitelzusammenfassung

In diesem Kapitel lag der Schwerpunkt auf der Entwicklung von Ansätzen des Repräsentationslernens zur Verarbeitung akustischer Daten.
Dazu wurde in Abschnitt 3.1 zunächst eine rekurrente Netzarchitektur zur Klassifikation von Primatenvokalisationen vorgestellt. Pro Mel-Spektrogramm werden mehrere Repräsentationen gelernt und diese anschließend mittels diverser Pooling-Mechanismen aggregiert. Die Architektur ist inspiriert durch das Vorgehen bei der manuellen Merkmalsextraktion, ist jedoch vollständig differenzierbar. Es konnte gezeigt werden, dass die Architektur eine deutlich bessere Genauigkeit erzielt als vergleichbare Ansätze und sich somit besonders für die Wildtierüberwachung eignet.
Im Gegensatz dazu behandelt das vorgestellte Verfahren aus Abschnitt 3.2 die akustische Leckerkennung, bei der zu den einzelnen Aufnahmen keine Labels vorliegen. Folglich kann die Problemstellung auch als Ein-Klassen-Klassifizierungsproblem aufgefasst werden. Es konnte gezeigt werden, dass neuronale Netze sich hierfür besonders gut eignen. Überdies konnte festgestellt werden, dass es ausreicht, einen Bruchteil der Aufnahmen auf anomale Geräusche zu überprüfen, um eine verlässliche Anomaliebwertung über einen festgelegten Entscheidungshorizont zu erhalten.
Ebenfalls zur Erkennung von Anomalien in akustischen Daten wurde in Abschnitt 3.3 DRINK vorgestellt. DRINK entfernt den mittleren Teil des Mel-Spektrogramms und sagt diese mit Hilfe eines rekurrent neuronalen Netzes vorher. Der Vorhersagefehler dient anschließend zur Berechnung der Anomaliebewertung. Anhand von Aufnahmen von Industriemaschinen konnte DRINK's Überlegenheit gegenüber vergleichbaren Ansätzen demonstriert werden.

4 Transferlernen zur akustischen Anomalieerkennung

In diesem Kapitel wird die in Abschnitt 3.2 und Abschnitt 3.3 bereits behandelte Problemstellung der akustischen Anomalieerkennung erneut aufgegriffen. Bei der Anomalieerkennung wird die Abweichung vom Normalfall quantifiziert. Ein verlässliches System zur Anomalieerkennung kann beispielsweise Betrug verhindern, Schäden vermeiden oder Krankheiten entdecken.
Dieses Kapitel folgt Kapitel 3 und widmet sich erneut der akustischen Anomalieerkennung in Geräuschen von Industriemaschinen wie Lüftern, Pumpen, Gleitschienen und Ventilen. Im Gegensatz zu den zuvor vorgestellten Ansätzen soll hier aber die Frage beantwortet werden, welchen Nutzen die Verwendung von vortrainierten neuronalen Netzen für die akustische Anomalieerkennung hat. Folglich liegt der Schwerpunkt dieses Kapitels auf den Auswirkungen des Wissenstransfers und nicht auf der Entwicklung von neuen Netzarchitekturen oder Optimierungszielen. Der Vorteil dieses Vorgehens ist, dass bereits trainierte und frei verfügbare neuronale Netze zur Merkmalsextraktion verwendet werden können, ohne dass sie von Grund auf neu entwickelt und trainiert werden müssen.
Abschnitt 4.1 untersucht zunächst die Eignung von Convolutional Neural Networks, die zuvor zur Bildklassifizierung trainiert wurden. Da zur Bildklassifizierung grundlegende Filter wie Kanten-, Textur- und Objektdetektoren erlernt werden, wird die Hypothese aufgestellt, dass sich semantisch sinnvolle Merkmale extrahieren lassen, die sich zur akustischen Anomalieerkennung eignen.
Um das Spektrum der Domänen, auf denen die neuronalen Netze zuvor trainiert wurden, zu erweitern, werden die Ergebnisse aus Abschnitt 4.1 in Abschnitt 4.2 verwendet, um die Bild-, Umweltgeräusch- und Musikdomäne miteinander zu vergleichen. Hier wird folglich genauer untersucht, ob Domänen, die strukturell ähnlicher zu den Geräuschen von Industriemaschinen sind, auch eine bessere Anomalieerkennungsleistung aufweisen.
Beide Abschnitte basieren in sequentieller Reihenfolge auf den Publikationen [Mül+21c] und [Mül+21a].
Abschnitt 4.3 fasst das Kapitel und dessen wichtigste Erkenntnisse kurz zusammen. Eine detailliertere Zusammenfassung der vorgestellten Ansätze sowie mögliche Ansatzpunkte für zukünftige Arbeiten finden sich am Ende eines jeden Abschnitts.

4.1 Transfer von Bildrepräsentationen

Verfahren der Anomalieerkennung identifizieren unerwartete Abweichungen von einem erwarteten Muster und werden zur Überwachung kritischer Infrastrukturen oder zur Erkennung von betrügerischem Nutzerverhalten eingesetzt. Die Mehrzahl der derzeitigen Ansätze basiert jedoch auf der Erkennung von Anomalien in Bildern.
Die visuelle Anomalieerkennung ist unzureichend, wenn die Umgebung nicht vollständig von Kameras erfasst werden kann, was zu blinden Flecken führt, in denen keine Vorhersagen gemacht werden können. Dies gilt vor allem auch für viele industrielle Produktionsanlagen und Maschinen. In vielen Fällen kann eine visuelle Inspektion den wahren Zustand der überwachten Einheit nicht erfassen. Eine Pumpe mit einer kleinen Leckage, eine Gleitschiene ohne Fett oder ein Lüfter, der Spannungsschwankungen unterliegt, können bei einer visuellen Inspektion intakt erscheinen, bei einer akustischen Überprüfung jedoch ihren tatsächlichen Zustand durch deutliche Geräuschmuster erkennen lassen. Darüber hinaus hat die akustische Überwachung den Vorteil, dass die Hardware vergleichsweise kostengünstig und leicht zu installieren ist. Die frühzeitige Erkennung von Maschinenfehlfunktionen mit einem zuverlässigen akustischen Anomalieerkennungssystem kann somit größere Schäden verhindern und die Reparatur- und Wartungskosten senken.
Mittlerweile sind CNNs in Verbindung mit dem Mel-Spektrogramm des Signals in der akustischen Signalverarbeitung allgegenwärtig. Typische Problemstellungen, die mit Hilfe von CNNs untersucht werden, umfassen die Klassifikation von Umweltgeräuschen [SB17], die Spracherkennung [Qia+16] oder das Musik-Tagging [PS19]. Jedoch beinhalten all diese Ansätze speziell auf die jeweilige Aufgabe zugeschnittene CNN-Architekturen und erfordern einen annotierten Datensatz. Im Kontext der akustischen Anomalieerkennung werden CNNs vor allem innerhalb von Autoencodern eingesetzt, bei denen der Rekonstruktionsfehler zur Anomaliebewertung dient. Im Gegensatz zu Convolutional Autoencodern wird beim klassischen Ansatz des Feature Engineering eine Reihe von manuell gewählten Merkmalen (die Fachwissen erfordern) aus dem vorliegenden Signal extrahiert und diese Merkmale werden als Eingabe für ein spezielles Anomalieerkennungsmodell wie z.B. einen Dichteschätzer verwendet. Die angesprochenen Modelle versagen jedoch bei hochdimensionalen Eingabedaten (z. B. Bilder, DNA-Sequenzen, Mel-Spektrogramme) aufgrund des Fluches der Dimensionalität.
In dieser Arbeit soll das Beste aus beiden Welten kombiniert und die Frage gestellt werden, ob es möglich ist, ein neuronales Netz zur automatischen Extraktion von Merkmalen zu verwenden. Diese Merkmale sollen in Verbindung mit traditionelleren Anomalieerkennungsmodellen eingesetzt werden und dabei eine vergleichbare oder sogar bessere Leistung erzielen.
Ausgehend von der Beobachtung, dass Muster anomaler Vorgänge oft visuell im Mel-Spektrogramm einer Aufnahme erkennbar sind, wird angenommen,

dass CNNs, die zuvor für die Bildklassifizierung trainiert wurden, nützliche Repräsentationen extrahieren können. Es gilt dann zu zeigen, dass die Repräsentationen aussagekräftig sind, obwohl sie für einen ganz anderen Zweck trainiert wurden.
Um Bilder korrekt zu klassifizieren, muss ein CNN zunächst grundlegende Filter wie Kanten-, Textur- und Objektdetektoren erlernen, die wiederum wertvolle und semantisch sinnvolle Merkmale extrahieren können, die sich auch auf verschiedene nachgelagerte Aufgaben übertragen lassen. Darüber hinaus verringert sich auf diese Weise der erforderliche Aufwand für die Suche nach einer geeigneten neuronalen Netzwerkarchitektur.

4.1.1 Stand der Forschung zum Transferlernen in verwandten Anwendungsbereichen

Verwandte Arbeiten, die hauptsächlich auf Autoencodern basieren, wurden bereits umfassend in Abschnitt 3.2.1 und Abschnitt 3.3.1 diskutiert. Bei diesen Ansätzen werden vor allem weitere Optimierungsziele oder Architekturen vorgeschlagen und für jede Problemstellung neu trainiert. Im Gegensatz dazu stützt sich der Ansatz in dieser Arbeit auf frei verfügbare und standardisierte CNN Architekturen, welche bereits trainiert wurden, gleichwohl nicht explizit für die akustische Anomalieerkennung. Es findet also kein erneutes Training des CNNs statt.
Die Umwidmung von vortrainierten CNNs wurde zunächst im Kontext der Klassifikation von Schnarchgeräuschen [Ami+17] vorgeschlagen. Dabei wurde gezeigt, dass die CNN basierten Repräsentationen die Leistung von wissensbasierten Merkmalen (6373 akustischen Funktionale) übertreffen. Später konnte die Effektivität der Repräsentationen auch für die audiobasierte Videospielgenre-Erkennung [Ami+20], die Erkennung von Hustenanfällen kranker Schweine [Yin+21] oder die Erkennung von Emotionen in der Sprache [Cum+17; Zha+18b] gezeigt werden.
Anstatt die Repräsentationen direkt zu extrahieren, verwenden vergleichbare Ansätze Fine-Tuning (Abschnitt 2.1.7) zur Klassifizierung von Herztönen [Ren+18; Dem+19]. Durch dieses Vorgehen werden die Gewichte des CNNs mit Hilfe eines annotierten Datensatzes noch weiter angepasst.
Abweichend von den besprochenen Ansätzen behandelt das in dieser Arbeit vorgestellte Verfahren die akustische Anomalieerkennung. In diesem Szenario sind zudem keine weiteren Labels zum Fine Tuning verfügbar. Weiterhin wird nicht nur eine einzige Kombination aus Netzarchitektur und Klassifizierer evaluiert, sondern über eine Vielzahl von Konfigurationen untersucht. Da die Dauer der Aufnahmen, für die eine Anomaliebwertung berechnet werden muss, in dieser Arbeit deutlich höher (10s) als in den anderen Ansätzen ist ($\approx 0.2 - 2.5$s), basiert der vorgeschlagene Ansatz zusätzlich auf einem Gleitfensterverfahren.

4.1.2 Ansatz zur Umwidmung von vortrainierten Netzen

Zum Lernen eines Modells des Normalfalls sei, wie auch schon in Gleichung 3.18, ein Datensatz $D = \{x_i \in \mathbb{R}^{F \times T} \mid 1 \leq i \leq n\}$ von Mel-Spektrogrammen gegeben. Da die Erhebung von Normalfalldaten in der Regel unproblematisch möglich ist, lassen sich die Aufnahmen in gleichgroße Segmente aufteilen, über die eine Anomaliebewertung berechnet werden soll.
Die Idee der vorliegenden Arbeit ist es, die Anomalieerkennung mit Hilfe der D dimensionalen Repräsentationen eines für die Klassifikation von Bildern vortrainierten $CNN : \mathbb{R}^{3 \times H \times W} \rightarrow \mathbb{R}^D$ (3 Farbkanäle, Höhe H, Breite W) durchzuführen.
Zunächst kann beobachtet werden, dass akustische Anomalien von Fabrikmaschinen oft visuell in den Mel-Spektrogrammen der Aufnahmen zu erkennen sind (siehe Abbildungen 3.6 und 3.10). Somit ist anzunehmen, dass ein CNN, das für Bilderkennung trainiert wurde, aussagekräftige Merkmale extrahieren kann, die helfen, zwischen normalem und anormalem Betrieb zu unterscheiden. Diese Annahme wird dadurch unterstützt, dass gezeigt werden konnte [OMS17; Ola+20], dass die Filter solcher Netze Farben, Kontraste, Formen (z.B. Linien, Kanten), Objekte und Texturen erkennen.
Das typische Eingabeformat für fast alle frei verfügbaren vortrainierten CNNs ist $H = W = 224$. Somit wird ersichtlich, dass das Format der Mel-Spektrogramme zunächst angepasst werden muss, um dem Definitionsbereich des CNNs zu entsprechen. Erschwerend hinzu kommt, dass der Zeithorizont, über den eine Anomaliebewertung berechnet werden soll, vom Einsatzszenario abhängt und von wenigen Sekunden bis zu vielen Stunden oder gar Tagen reichen kann. Entsprechend wächst die zeitliche Dimension T eines jeden x_i.
Ein einfacher und weit verbreiteter Ansatz ist es nun, die Spektrogramme in RGB-Bilder umzuwandeln und so zu strecken und zu stauchen, dass das resultierende Bild die Form $3 \times 224 \times 224$ annimmt. In fast allen der in Abschnitt 4.1.1 vorgestellten Arbeiten wird dieser Ansatz verfolgt. In deren Fall ist dieses Vorgehen noch möglich, da es sich um kurze Aufnahmen handelt, bei denen durch die Streckung und Stauchung nur wenig Informationen verloren gehen. Wächst T jedoch stark an, dann wird das Verhältnis zwischen F und T so unausgeglichen, dass ein quadratisches Bild keine geeignete Darstellung mehr ist. Wenn beispielsweise $T >> 224$ ist, dann sind nur kurz andauernde akustische Ereignisse nicht mehr sichtbar.
In Anlehnung an das Verfahren aus Abschnitt 3.2.4.1 wird vorgeschlagen einen festen Zeitabschnitt zu wählen, für den ein RGB-Bild des Mel-Spektrogramm Ausschnitts erstellt wird. In dieser Arbeit wird das zugrundeliegende Mel-Spektrogramm in eine Sequenz von quadratischen ($T = F$) Abschnitten aufgeteilt. Somit muss das Mel-Spektrogramm bei der Konvertierung in ein RGB-Bild lediglich hochskaliert werden.

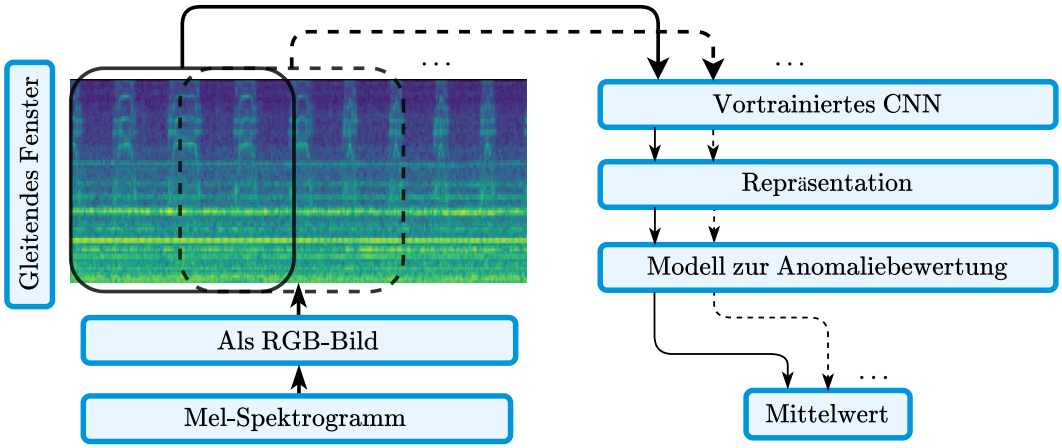


Abbildung 4.1: Überblick über das vorgeschlagene Verfahren. Zunächst werden mit Hilfe eines gleitenden Fensters kleine quadratische Segmente (~ 2s) extrahiert. Nachfolgend wird ein für die Bildklassifizierung bereits vortrainiertes neuronales Netz verwendet, um Repräsentationen für die Segmente zu extrahieren. Die Repräsentationen dienen als Eingabe für ein Anomalieerkennungs-Modell. Eine Vorhersage über die gesamte Aufzeichnung wird durch Mittelwertbildung der einzelnen Anomaliebewertungen berechnet.

4.1.3 Versuchsaufbau und Bewertung der Leistungsfähigkeit des Ansatzes

Um die Wirksamkeit des vorgestelleten Ansatzes für die Erkennung akustischer Anomalien zu untersuchen, wird wieder der in Abschnitt 3.3.3.1 vorgestellte MIMII Datensatz verwendet [Pur+19]. Alle Aufnahmen im Datensatz werden zunächst zu Mel-Spektrogrammen unter Verwendung von 64 Mel-Bändern konvertiert. Anschließend werden Ausschnitte der Größe 64×64 (~ 2s) mit einem gleitenden Fenster mit Schrittlänge 32 (~ 1s) entlang der Zeitachse extrahiert und in 224×224 große RGB-Bilder konvertiert. Dabei kommt, wie in [Ami+17] empfohlen, die *viridis* Farbskala zum Einsatz[1]. Die Bilder werden mit dem Mittelwert und der Standardabweichung des Datensatzes, auf dem das CNN ursprünglich trainiert wurde [Den+09; KSH12], normalisiert. Aufgrund der gewählten Größe der Mel-Spektrogramm-Segmente bleibt das ursprüngliche Seitenverhältnis unverändert, was einem möglichen Informationsverlust entgegenwirkt. Schließlich werden verschiedene vortrainierte CNNs eingesetzt, um eine Repräsentation für jedes Bild zu generieren. Auf Basis dieser Repräsentationen werden nun wiederum verschiedene Modelle zur Anomalieerkennung trainiert. Eine Anomaliebewertungen für eine einzelne Aufnahme

[1] Für das vorliegende Anwendungsszenario konnte festgestellt werden, dass die Wahl der Farbskala keinen Einfluss auf die Leistungsfähigkeit des Ansatzes hat.

entsteht dann durch den Mittelwert der Anomaliebewertungen der einzelnen Ausschnitte.

4.1.3.1 Vortrainierte CNNs

Die folgenden vortrainierten CNNs wurden zunächst auf dem ImageNet Datensatz [Den+09] vortrainiert. Er enthält insgesamt 1000 Klassen mit mehreren Millionen Bildern wie z.B. Tieren, Gesichtern oder Objekten.

i) AlexnetV3 [KSH12]: Die CNN Architektur, die Feauture-Engineering-Ansätze zum ersten Mal deutlich übertroffen hat. Die Filtergröße reduziert sich mit steigender Anzahl von Schichten (11×11, 5×5 und 3×3). Max-Pooling wird alle drei Schichten eingesetzt, um die Größe der Feature-Maps zu reduzieren. Ein FFNN mit zwei versteckten Schichten berechnet die Klassenzugehörigkeit. Die Aktivierungen der vorletzten Schicht des FFNN bilden die 4096 dimensionale Repräsentation.

ii) ResNet18 [He+16]: Die am weitesten verbreitete CNN Architektur wurde entwickelt, um dem Problem des abnehmenden Nutzens bei zunehmender Netztiefe zu begegnen. Indem mehrere Schichten jeweils zu Blöcken zusammengefasst werden und eine Residualverbindung ($\text{block}(x) + x$ anstelle von $\text{block}(x)$) zwischen Blockeingang und Blockausgang gelegt wird, wird das Problem des verschwindenden Gradientens abgeschwächt und die Netzwerktiefe (hier 18 Schichten) kann weiter erhöht werden. Bis auf die erste Schicht, in der 7×7 Filter eingesetzt werden, werden ausschließlich 3×3 Filter verwendet. Die Aktivierungen der vorletzten Schicht bilden die 512 dimensionale Repräsentation.

iii) ResNet34 [He+16]: Folgt den gleichen Grundsätzen wie ResNet18 bei einer erhöhten Tiefe von insgesamt 34 Schichten und einer 2048-dimensionalen Repräsentation.

iv) SqueezeNet [Ian+16]: Diese Architektur wurde entwickelt, um so wenige Parameter wie möglich zu verwenden (50 mal weniger als AlexNet) und dennoch eine vergleichbare Klassifizierungsgenauigkeit zu erreichen. Dies wird mit Hilfe von *Fire*-Schichten erreicht, die mit *Squeeze* (1×1) und *Expand* (3×3) Modulen ausgestattet sind. Auf die finale Feature Map vor der Ausgabeschicht wird 2×2 Average-Pooling angewendet, um eine 2048-dimensionale Repräsentation zu berechnen.

4.1.3.2 Anomalieerkennungsmodelle

Zur Berechnung einer Anomaliebewertung für einzelne Bilder werden dieselben etablierten [Ped+11] Dichteschätzer und Ensemblemodelle aus Abschnitt 3.2.4.2 (akustische Leckerkennung) verwendet.
Zum Vergleich mit dem weit verbreiteten Ansatz der Anomalieerkennung auf

Grundlage des Rekonstruktionsfehlers werden darüber hinaus die folgenden Convolutional Autoencoder evaluiert:

i) DCAE: Dieser Autoencoder besitzt in seinen drei Schichten $32, 64$ und 128 Filter der Größe 5×5 und verwendet die Exponential Linear Unit [CUH16]. Zwischen jeder Schicht wird Batch Normalization [IS15] eingesetzt.

ii) DCAE (klein): Eine einfachere und verkleinerte Architektur mit $12, 24$ und 48 Filtern der Größe 4×4 und der ReLU [NH10] Aktivierungsfunktion.

Die Decoder spiegeln die genannten Schichten unter Verwendung von Deconvolutions. Das Training verläuft analog zu Abschnitt 3.2.4.1.

4.1.3.3 Bewertung der Leistungsfähigkeit des Ansatzes

In diesem Abschnitt werden die wichtigsten Erkenntnisse und Ergebnisse diskutiert. Die Erkenntnisse sind dabei in vier klare Aussagen gegliedert. Es gilt zu beachten, dass sich diese Aussagen nur auf das Anwendungsszenario dieses Abschnitts beziehen.

i) Der Transfer von Repräsentation von vortrainierten CNNs ist effektiver als von Grund auf neu trainierte Autoencoder. Autoencoder übertreffen die Kombination aus CNN und Anomalieerkennungsmodell nur in einer einzigen Konfiguration (DCAE-klein auf Pumpe-6). Dabei liefert jedoch in der Mehrzahl der Fälle der DCAE bessere Ergebnisse als die kleinere Variante. Meistens kommen die DCAEs nicht an die Ergebnisse ihrer Konkurrenten heran, wodurch die Hypothese unterstützt wird, dass die Merkmale, die durch gelernte Filter aus vortrainierten Bildklassifizierungsmodellen extrahiert werden, besser für die Erkennung von akustischen Anomalien geeignet sind.

ii) ResNet-Architekturen eignen sich besonders gut zur Extraktion von Repräsentationen. Um die Merkmalsextraktoren zu vergleichen, werden die Fälle gezählt, in denen ein bestimmtes CNN in Kombination mit verschiedenen Anomalieerkennungsmodellen die höchste oder zweithöchste durchschnittliche AUC erzielt, wobei Tupel der Form (erster, zweiter) gebildet werden. Wie in Tabelle 4.1 dargestellt, bestehen 16 verschiedene Szenarien, bei denen entweder die höchste oder die zweithöchste Punktzahl erreicht werden kann.
In der Reihenfolge vom besten bis zum schlechtesten Ergebnis ergibt sich folgendes Gesamtresultat: ResNet34 $(7, 6)$, ResNet18 $(3, 5)$, AlexNet $(3, 2)$, SqueezeNet $(2, 2)$ und Autoencoder $(1, 0)$. Hierbei ist eine eindeutige Überlegenheit von ResNet-basierten Repräsentationen zu beobachten. Interessanterweise sind dies auch die Modelle mit einem geringeren Klassifikationsfehler auf ImageNet. Diese Ergebnisse decken sich mit einer kürzlich erfolgten Beobachtung, dass eine starke Korrelation zwischen der Top-1-Genauigkeit von ImageNet und der Güte des Transfers der Repräsentationen

		Lüfter				Pumpe				Gleitschiene				Ventil			
		M0	M2	M4	M6	M0	M2	M4	M6	M0	M2	M4	M6	M0	M2	M4	M6
AlexNet	GMM	57.7	61.7	53.9	**94.5**	84.1	70.8	81.6	66.0	98.3	80.9	61.4	57.5	60.2	69.2	59.9	53.5
	B-GMM	50.9	61.4	47.7	82.2	71.8	60.2	73.4	53.3	83.2	65.0	50.0	57.0	55.2	62.7	51.4	48.3
	IF	53.1	59.7	48.9	84.6	75.9	62.4	75.0	55.9	89.4	69.0	51.9	56.2	50.1	63.4	53.3	49.8
	KDE	55.7	59.1	50.5	90.3	76.4	65.9	74.8	61.0	97.8	79.3	59.7	55.0	54.6	64.4	57.1	51.4
	OC-SVM	51.0	**73.1**	**59.7**	93.2	77.5	56.4	81.1	60.1	96.2	81.4	53.6	56.5	61.6	73.6	48.3	48.9
ResNet18	GMM	**62.6**	64.1	59.3	94.4	**84.5**	**71.3**	84.0	68.3	99.1	85.8	68.8	65.6	58.3	73.3	60.2	56.9
	B-GMM	59.2	60.5	54.8	91.0	79.1	69.7	79.4	59.5	98.3	77.7	61.4	61.2	70.1	71.7	56.1	50.3
	IF	58.0	60.5	55.3	86.5	70.8	59.0	77.3	54.6	97.7	72.7	60.6	61.2	56.5	69.8	58.2	47.5
	KDE	57.9	59.1	55.6	85.9	76.6	56.5	76.7	62.2	98.1	77.0	61.2	60.9	57.6	62.9	56.8	49.7
	OC-SVM	55.0	68.8	57.4	87.7	71.6	55.2	78.6	60.6	96.7	79.6	69.3	66.2	61.1	76.1	56.8	43.1
ResNet34	GMM	58.7	65.6	57.0	90.9	78.4	66.8	87.9	63.2	**99.6**	**90.4**	**82.5**	69.1	73.0	**79.1**	60.1	**61.9**
	B-GMM	55.7	61.8	52.3	85.8	71.5	61.1	84.5	55.2	99.2	85.4	72.3	63.6	70.8	76.2	59.3	57.9
	IF	53.9	62.0	49.9	82.2	52.3	48.3	79.3	49.4	98.6	83.1	69.5	60.2	65.9	71.2	60.3	54.0
	KDE	55.0	62.6	52.3	83.1	62.0	51.8	82.8	58.3	99.0	84.0	68.2	62.2	67.5	71.9	53.9	58.2
	OC-SVM	50.1	67.4	57.5	83.0	64.9	51.5	81.2	60.2	96.8	85.0	71.4	64.3	**75.6**	77.8	**64.3**	53.1
SqueezeNet	GMM	56.1	60.4	49.4	83.4	72.1	46.4	87.6	60.8	96.7	76.8	52.1	62.9	62.8	75.3	53.3	57.3
	B-GMM	54.4	59.8	47.0	84.5	72.3	48.2	86.2	69.0	95.0	78.8	55.8	65.0	63.8	74.0	52.4	56.8
	IF	53.2	64.0	44.8	84.6	76.1	45.5	85.3	60.2	98.9	78.2	53.1	70.6	56.6	68.7	51.5	56.6
	KDE	54.4	60.5	47.0	84.3	74.5	45.2	86.5	61.4	98.7	80.8	56.4	69.2	65.0	74.5	52.8	57.7
	OC-SVM	55.6	64.8	46.2	86.7	78.8	49.4	**88.4**	62.3	99.2	81.5	59.4	**71.6**	69.0	71.3	53.1	58.2
DCAE	-	49.1	57.0	53.2	66.9	65.3	54.4	76.0	66.6	95.9	70.4	56.2	50.6	42.3	55.6	51.2	45.5
DCAE (klein)	-	48.3	54.1	49.3	63.7	69.9	52.9	73.1	**69.2**	95.3	68.4	55.7	53.3	36.6	57.2	51.2	45.4

Tabelle 4.1: Leistungsvergleich der Kombination aus vortrainiertem CNN und Anomalieerkennungs-Modell. Die beste Kombination pro Datensatz ist fett geschrieben, die zweitbeste unterstrichen.

besteht [KSL19].
Ebenfalls wichtig ist die Tatsache, dass die gute Leistung von ResNet34 fast ausschließlich aus dem guten Abschneiden bei Gleitschienen und Ventilen resultiert. Die Mel-Spektrogramm-Bilder dieser Maschinen weisen weitaus feingranularere Variationen auf als die von Lüftern und Pumpen, welche eine stationärere Belegung der einzelnen Frequenzbänder aufweisen. Es ist anzunehmen, dass ResNet34 Merkmale auf einer detaillierteren Ebene extrahiert, was die geringere Leistung bei Aufnahmen von Lüftern und Pumpen erklären kann. Im Allgemeinen erwiesen sich die Repräsentationen von SqueezeNet als die am wenigsten zuverlässigen.

3) GMM und OC-SVM liefern die besten Ergebnisse. Um die Modelle zur Erkennung von Anomalien zu vergleichen, werden die Fälle gezählt, in denen ein bestimmtes Modell zur Erkennung von Anomalien in Kombination mit verschiedenen Merkmalsextraktoren das beste oder zweitbeste Ergebnis erzielt. Bei Anwendung der gleichen Ranking-Strategie wie zuvor erhält man die folgenden Ergebnisse: GMM $(9, 8)$, OC-SVM $(6, 2)$, Autoencoder $(1, 0)$, B-GMM $(0, 3)$, IF $(0, 2)$, KDE $(0, 0)$. Die Ergebnisse zeigen deutlich, dass GMM und OC-SVM alle anderen Modelle klar übertreffen. Gemeinsam sind sie für 15/16 der leistungsfähigsten Modelle und 10/16 der zweitleistungsfähigsten Modelle verantwortlich. Obwohl GMM und B-GMM auf denselben theoretischen Annahmen beruhen, liefert B-GMM schlechtere Ergebnisse. Dies ist vermutlich darauf zurückzuführen, dass die A-Priori Verteilung der Parameter der B-GMM zu restriktiv ist.

4) Die Ergebnisse sind in hohem Maße vom Maschinentyp und vom Maschinenmodell abhängig. Das Modell, das bei Ventilen am besten abschneidet, hat eine durchschnittliche AUC von 79.1. Dieser Wert ist im Vergleich zu den anderen Maschinentypen sehr niedrig, da diese stets mindestens ein Szenario mit einer mittleren AUC > 80 aufweisen. Zudem variiert die höchste erreichte AUC über alle Maschinentypen hinweg erheblich. Daraus lässt sich ableiten, dass einige Maschinentypen für den Ansatz besser geeignet sind (Pumpen, Gleitschienen) als andere (Lüfter, Ventile). Noch wichtiger ist jedoch, dass eine erhebliche Varianz zwischen den verschiedenen Maschinenidentifikatoren (M0 - M6) zu beobachten ist. Die Ergebnisse für Ventilatoren verdeutlichen dieses Problem am besten. Während M0, M2 und M4 Durchschnittswerte von $62.6, 73.1$ und 59.7 aufweisen, erreicht M6 einen Durchschnitt von 94.5. M6 verbessert sich gegenüber M4 um $\sim 30\%$. Demnach sind anomale Klangmuster selbst bei verschiedenen Modellen desselben Maschinentyps sehr unterschiedlich (mehr oder weniger subtil).

4.1.4 Fazit und Diskussion möglicher Verbesserungen

In diesem Abschnitt wurde die Erkennung akustischer Anomalien von Maschinengeräuschen untersucht. Für die Extraktion von Repräsentationen wurden vier frei verfügbare CNNs verwendet, die zuvor für die Klassifizierung von ImageNet-Bildern trainiert wurden.
Mit diesen Repräsentationen wurden anschließend fünf verschiedene Anomalieerkennungsmodelle trainiert. Die Ergebnisse zeigen eine klare Überlegenheit von ResNets in Kombination mit einem GMM oder einer OC-SVM.
Ferner bestätigte sich die Hypothese, dass bildbasierte Repräsentationen vielseitig einsetzbar sind und folglich auch konkurrenzfähige Ergebnisse bei der Erkennung akustischer Anomalien liefern.
Zukünftige Arbeiten könnten Ensemble-Ansätze und andere CNN Architekturen untersuchen [How+17; Hua+17; Kaw+19; PS19]. Darüber hinaus könnte der Ansatz von Techniken zur Reduktion von Hintergrundrauschen [Zha+18a] profitieren. Zudem wäre es denkbar, vortrainierte Merkmalsextraktoren aus anderen, verwandten Domänen wie Musik oder Umweltgeräuschen zu verwenden. Auch die Evaluation von CNNs, die zur Objekterkennung, Objektsegmentierung oder Instanzsegmentierung trainiert wurden, wäre möglich.
Da in dieser Arbeit bereits gezeigt wurde, dass die untersuchten CNNs gute Transferleitungen liefern, obwohl deren eigentliche Aufgabe eine ganz andere ist, stellt sich die Frage, welche Faktoren zu diesem Ergebnis beitragen. Da neuronale Netze *Black-Boxen* sind, lassen sich mit dem aktuellen Stand der Forschung [Zha+21b] nur sehr eingeschränkt Aussagen darüber treffen, wie die Repräsentationen berechnet werden und welche Komponenten der CNNs gute Repräsentationen für die akustische Anomalieerkennung begünstigen.
Das vorgestellte Verfahren wurde bewusst einfach gehalten, um den Fokus auf den einzelnen vortrainierten CNNs zu belassen, sodass so wenig zusätzliche Module wie möglich die Ergebnisse beeinflussen. Durch die Verwendung von RNNs oder ausgefeilteren Aggregationsstrategien ist davon auszugehen, dass die Leistungsfähigkeit noch verbessert werden kann.

4.2 Transfer von Repräsentationen aus unterschiedlichen Domänen

Das Gebiet der Erkennung anomaler Geräusche bildet das Gegenstück zur Erkennung von Anomalien in Bilddaten und widmet sich Situationen, in denen eine visuelle Überwachung nicht möglich oder deutlich aufwändiger ist. Dabei ist einer der wichtigsten Anwendungsfälle die frühzeitige Erkennung von Fehlfunktionen während des Betriebs von Fabrikmaschinen.
Bei dem weit verbreiteten, auf Autoencodern basierendem, Ansatz müssen die Modelle jedes Mal von Grund auf neu trainiert werden. Darüber hinaus nutzen sie weder Vorwissen noch zusätzliche Trainingsdaten aus anderen Domänen aus. In Abschnitt 4.1 wurde daraufhin ein Ansatz basierend auf vortrainierten

CNNs vorgestellt, der es ermöglicht, das bereits in diesen Netzen enthaltene Wissen für die akustische Anomalieerkennung auszunutzen. Das Klassifikationsproblem auf welchem die CNNs zuvor trainiert wurden, hat jedoch kaum Gemeinsamkeiten mit der nachfolgenden Aufgabe der Erkennung anomaler Geräusche.
Aufgrund der weiten Verfügbarkeit von akustischen Datensätzen und neuronalen Netzen, die für verwandte Aufgaben wie der Klassifizierung von Umgebungsgeräuschen trainiert wurden, stellt sich die Frage, ob deren Wissen übertragen und genutzt werden kann, um die Erkennungsleistung zu verbessern. Dieser Frage soll in der vorliegenden Arbeit nachgegangen werden.
Dazu werden Repräsentationen aus drei verschiedenen Domänen, die aufgrund ihrer strukturellen Ähnlichkeit mit Maschinengeräuschen ausgewählt wurden, evaluiert: i) Bildbasierte Repräsentationen, ii) Umweltgeräuschbasierte Repräsentationen, iii) Musikbasierte Repräsentationen. Aufbauend auf den Ergebnissen aus Abschnitt 4.1 wird ein Gaussian Mixture Model für die Berechnung der Anomaliebewertungen verwendet. Zudem wird gezeigt, dass alle Repräsentationen die Autoencoder-Baseline [Kaw+19] übertreffen und musikbasierte Repräsentationen die besten Ergebnisse liefern.

4.2.1 Überblick über die Verwendung des Transferlernens in anderen Forschungsfeldern

Transferlernen (Abschnitt 2.1.7) wird hauptsächlich in der Computervision und der natürlichen Sprachverarbeitung (NLP) eingesetzt, hat aber in letzter Zeit auch in der akustischen Signalverarbeitung an Bedeutung gewonnen.
In der Computervision wird dabei in den meisten Fällen auf ImageNet vortrainierte CNNs zurückgegriffen.
Im NLP hat das Aufkommen von vortrainierten Sprachmodellen zu erheblichen Verbesserungen bei einer Vielzahl von Aufgaben, wie z.B. der Sentimentanalyse oder der maschinellen Übersetzung, geführt. Fast alle derzeitigen Ansätze verwenden zunächst vortrainierte Repräsentationen für einzelne Wörter [Mik+13c] und im Anschluss vortrainierte Transformer [Dev+19] oder RNNs [Pet+18b], die auf der maskierten Sprachmodellierung basieren.
Auch auf dem Gebiet der akustischen Signalverarbeitung existieren mittlerweile einige vortrainierte Netze. Beckmann et al. [Bec+19] verwenden die klassische VGG-16-Architektur [SZ15] zur Klassifizierung gesprochener Wörter und zeigen, dass sich die Repräsentationen auch für die Sprachverbesserung, Spracherkennung sowie die Klassifizierung von Sprache, Geräuschen und Musik, eignen. Cramer et al. [Cra+19] formulieren das Optimierungsziel als selbstüberwachte audiovisuelle Korrespondenzaufgabe und verwenden die resultierenden Repräsentationen, um die Klassifikationsgenauigkeit von Umgebungsgeräuschen zu verbessern. Andere Ansätze [Kon+20; Gon+22] basieren auf dem Training von neuronalen Klassifikatoren auf AudioSet [Gem+17]. AudioSet ist ein umfangreicher Datensatz mit manuell annotierten Aufnahmen, der

menschliche-, tierische- und Umweltgeräusche sowie Musikstücke enthält.
Obwohl diese Netze frei verfügbar sind, werden sie in der Praxis nur selten verwendet, d.h. die meisten Modelle werden von Grund auf neu trainiert. Die möglichen Vorteile, welche die Einbeziehung von vortrainierten Netzen haben könnte, werden somit vernachlässigt.
Keine der angesprochenen Arbeiten betrachtet als nachgelagerte Aufgabe die akustische Anomalieerkennung oder evaluiert die Auswirkungen der Trainingsdomäne.

4.2.2 Ansatz zum Transfer von Repräsentationen

Dem Ansatz liegt die Idee zugrunde, vortrainierte neuronale Netze zur Extraktion von geeigneten Repräsentationen zu nutzen. In dieser Arbeit werden dabei neuronale Netze betrachtet, die Merkmale auf Basis des Mel-Spektrogramms einer Tonaufnahme berechnen. Der Definitionsbereich der Netze ist dabei bereits passend zu dem initialen Optimierungsziel gewählt worden und folglich bereits festgelegt. In der Regel nehmen die Netze Mel-Spektrogramme von Tonaufnahmen von ca. 0.5s bis 3s entgegen. Ist die Tonaufnahme länger, im vorliegenden Fall 10s, so kann mit Hilfe eines gleitenden Fensters mit einer bestimmten Schrittlänge eine Sequenz von Repräsentationen extrahiert werden. Sei diese Sequenz gegeben als Matrix aus Spaltenvektoren $\mathbb{R}^{D \times T}$:

$$R = \begin{bmatrix} | & | & | \\ r_1 & \cdots & r_T \\ | & | & | \end{bmatrix} \tag{4.1}$$

Eine Anomaliebewertung für R soll nun auf Basis eines GMMs berechnet werden. Wie schon in Abschnitt 4.1.3.3 festgestellt wurde, eignen sich GMMs besonders gut für die akustische Anomalieerkennung mit vortrainierten neuronalen Netzen. Dazu sei zunächst

$$p(r_1, \ldots, r_T) = \prod_{t=1}^{n} p(r_t \mid r_1 \ldots, r_{t-1}) \tag{4.2}$$

Die Wahrscheinlichkeit einer Repräsentation zum Zeitpunkt t hängt nach Gleichung 4.2 somit von allen vorangegangen Repräsentationen ab. Zur Vereinfachung des Modells wird diese Abhängigkeit nicht berücksichtigt, d.h. es wird bedingte Unabhängigkeit angenommen, sodass die Wahrscheinlichkeit der Sequenz von Repräsentationen wie folgt gegeben ist:

$$p(r_1, \ldots, r_n) = \prod_{t=1}^{n} p(r_t) \tag{4.3}$$

Das GMM führt nun die zusätzliche Annahme ein, dass alle Repräsentationen aus einer Affinkombination einer endlichen Anzahl von Gaußverteilungen mit

unbekannten Parametern erzeugt werden.
Die Wahrscheinlichkeit für eine Repräsentation ist folglich gegeben durch

$$p\left(r_t \middle| \pi, \mu, \Sigma\right) = \prod_{k=1}^{K} \pi_k \mathcal{N}\left(r_t \middle| \mu_k, \Sigma_k\right) \quad \text{wobei} \quad \sum_{k=1}^{K} \pi_k = 1 \tag{4.4}$$

Die unbekannten Parameter der Gaußerverteilungen $\Sigma = \{\Sigma_1, \dots \Sigma_K\}$ und $\mu = \{\mu_1, \dots \mu_K\}$ sowie die Gewichtungskoeffizienten $\pi = \{\pi_1, \dots \pi_K\}$ für jede der K Verteilungen werden dabei auf Basis der Menge aller extrahierten Repräsentationen geschätzt.
Genauer schätzt ein GMM die Wahrscheinlichkeitsverteilung aller Repräsentationen durch die Kombination von einfachen Gaußverteilungen. Dadurch lassen sich auch nicht normalverteilte, multimodale Wahrscheinlichkeitsverteilungen modellieren. Da die Repräsentationen der vortrainierten Netze in der Regel nicht regularisiert werden, um einer speziellen Verteilung zu folgen, ist die Flexibilität des GMMs im vorliegenden Anwendungsfall ein wesentlicher Vorteil.

4.2.3 Evaluation der Konkurrenzfähigkeit des Ansatzes

In diesem Abschnitt werden zunächst die zur Extraktion der Repräsentationen verwendeten neuronalen Netze genauer besprochen. Darauf folgt die Vorstellung des Aufbaus der Experimente sowie eine kurze Diskussion darüber, wie die Hyperparameter des Gaussian Mixture Models gewählt wurden. Schließlich folgt die Diskussion der Ergebnisse.
Zur Evaluation wird erneut der bereits in Abschnitt 3.3.3.1 vorgestellte MIMII Datensatz verwendet. Anders als in Abschnitt 3.11 und Abschnitt 4.1.3.3 werden im Folgenden jedoch alle Signal-zu-Rausch-Verhältnisse (−6dB, 0dB und 6dB) berücksichtigt.

4.2.3.1 Vortrainierte neuronale Netze

Die folgenden neuronalen Netze wurden auf Basis von Umweltgeräuschen trainiert. Die Domäne ist strukturell mit Maschinengeräuschen verwandt.

i) VGGish [Her+17b]: VGGish ist eine Variante der VGG-Architektur, die zuvor bereits erfolgreich zur Bildklassifizierung eingesetzt wurde. Im Vergleich zu VGG16 wurde die Anzahl der Schichten auf 11 und die Repräsentationsgröße auf 128 reduziert. Das Netz wurde für die Klassifizierung von ca. 5.24Mio. Stunden an Umweltgeräuschen trainiert.

ii) L3env [Cra+19]: Dieses Netz verwendet eine simple CNN Architektur [AZ17] mit 3×3 Filtern und insgesamt acht Schichten mit steigender Anzahl an Filtern. Das Trainingsziel ist das Lösen einer audiovisuellen Korrespondenzaufgabe, d. h. die Vorhersage, ob ein Mel-Spektrogramm

die Tonspur eines Videos wiedergibt oder nicht. Somit werden bei diesem Verfahren keine Annotationen benötigt. Das Training wurde mit ca. 195000 Videos (10s) mit Umweltgeräuschen in der Tonspur trainiert.

Musikbezogene Aufgaben wie das Musik-Tagging stützen sich auf eine Vielzahl von Faktoren wie Tonlage, Klangfarbe und Tonhöhe. Diese Merkmale können auch für die akustische Anomalieerkennung von Bedeutung sein. Die beiden nachstehenden neuronalen Netze wurden für musikbezogene Aufgaben trainiert:

i) MusiCNN [PS19]: Die Filterformen der CNN basierten Architektur sind explizit darauf ausgelegt, Phänomene und Muster zu berücksichtigen, die typischerweise in der Musik auftreten (z. B. Tonhöhe, Klangfarbe, Tempo). Breitere Filter modellieren längere zeitliche Abhängigkeiten und höhere Filter werden verwendet, um die Klangfarbe besser zu erfassen. MusiCNN wurde auf dem MagnaTagATune-Datensatz [Law+09] (ca. 26000 29s lange Musikstücke) für das Musik-Tagging trainiert.

ii) L3music [Cra+19]: Der einzige Unterschied zu L3env besteht darin, dass L3music mit ca. 296000 Videos von Personen trainiert wurde, die ein Musikinstrument spielen.

Aufgrund der Beobachtung, dass anomale Klangmuster oft visuell in ihrer spektralen Darstellung erkannt werden können, eignen sich für die vorliegende Problemstellung auch neuronale Bildklassifikatoren. Dies wird ebenfalls bei der Begutachtung von Abbildung 3.10 deutlich. Der Ansatz wurde bereits in Abschnitt 4.1 vorgestellt. Zusätzlich wird jedoch eine in Abschnitt 4.1.3.1 noch nicht berücksichtigte Netzarchitektur untersucht.

i) ResNet34 [He+16]: ResNet ist die am weitesten verbreitete CNN Architektur und wurde bereits in Absschnitt 4.1.3.1 vorgestellt. In Abschnitt konnte darüber hinaus gezeigt werden, dass das ResNet34 zu den leistungsfähigsten vortrainierten CNNs für die akustische Anomalieerkennung gehört.

ii) DenseNet121 [Hua+17]: Genau wie das ResNet wurde das DenseNet entwickelt, um eine größere Netzwerktiefe zu ermöglichen und gleichzeitig weniger Parameter zu benötigen. Hier erhält die Folgeschicht die Feature Maps aus allen vorherigen Schichten und berechnet, entsprechend einer Wachstumsrate, eine geringe Anzahl neuer Feature Maps. Dieser Prozess wird mit 1×1-Convolutions und Average-Pooling kombiniert, um die Anzahl und Größe der Feature-Maps zu reduzieren.

Zum Vergleich mit dem weit verbreiteten Autoencoder Ansatz dient die Architektur und das Trainingsverfahren von Koizumi et al. [Koi+20b]. Verglichen mit der Architektur aus Abschnitt 3.3.3.3 verwendet dieser Autoencoder

Name	Trainingsdomäne	Größe
L3Env	Umweltgeräusche	512
VGGish		128
ResNet34	Bilder	512
DenseNet121		1024
MusiCNN	Musik	753
L3music		512

Tabelle 4.2: Überblick über die verwendeten neuronalen Netze.

weniger Neuronen und Mel-Bänder. Diese Entscheidung wurde getroffen, um ausschließlich bewährte Architekturen aus der Literatur miteinander zu vergleichen und eine zusätzliche Untersuchung der Autoencoder-Architekturen zu vermeiden.
Ein Überblick über die verschiedenen Merkmalsextraktoren sowie die zugehörigen Trainingsdomänen und Repräsentationsgrößen sind in Tabelle 4.2 zu finden.

4.2.3.2 Vorgehen

Um die Eignung der verschiedenen Merkmalsextraktoren zu untersuchen, wird wie folgt verfahren:

i) Berechne die Repräsentationen für jede 10s lange Aufnahme im Datensatz mit einer Fenstergröße von 1s und einer Schrittweite von 0.5s. Dadurch ergeben sich 20 Merkmalsvektoren pro Aufnahme.

ii) Standardisiere die Repräsentationen und reduziere deren Dimensionalität mittels Hauptkomponentenanalyse, sodass 98% der Varianz erhalten bleibt.

iii) Nutze die angepassten Repräsentationen zum Training eines GMMs.

iv) Zum Testzeitpunkt werden die angepassten Repräsentationen für jede 10s lange Aufnahme (Schritte 1 und 2) im Testdatensatz bestimmt. Die Anomaliebewertung für eine Repräsentation ergibt sich aus den gewichteten negativen log-Wahrscheinlichkeiten. Durch Mittelwertbildung ergibt sich die endgültige Anomaliebewertung.

Die Ergebnisse der einzelnen Experimente sind in Tabelle 4.3 zu finden. Um einen besseren Überblick über die Verteilung der Leistung der einzelnen vortrainierten Netze zu erhalten, wird in den Abbildungen 4.3, 4.4 und 4.5 zwischen den Trainingsdomänen unterschieden und über alle Maschinenidentifikatoren aggregiert. Die besten Ergebnisse pro Trainingsdomäne werden dann in Abbildung 4.6 miteinander verglichen. In Tabelle 4.4 werden schlussendlich die besten Ergebnisse aus Tabelle 4.3 übernommen und mit DRINK aus

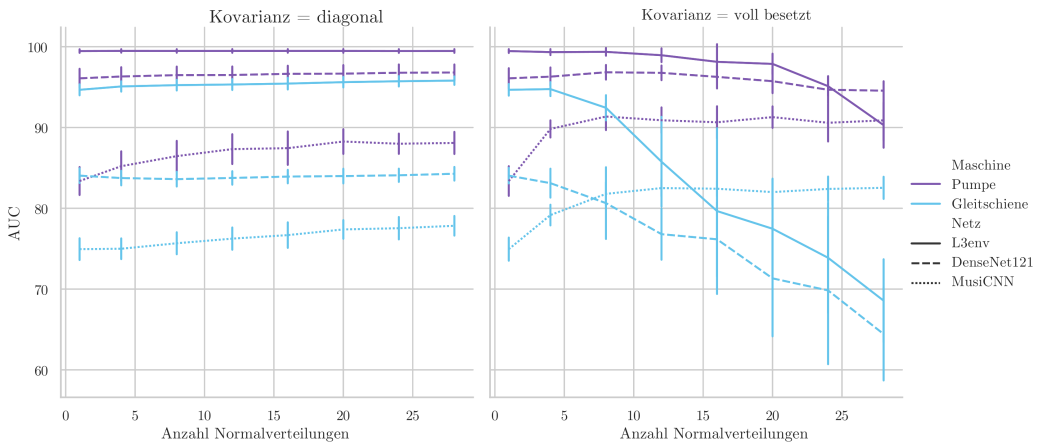


Abbildung 4.2: Evaluation der GMM Hyperparameter.

Abschnitt 3.3 verglichen (Signal-Rausch-Verhältnis von 0dB).

Die entscheidenden Hyperparameter des GMMs sind die Anzahl der multivariaten Normalverteilungen und der Typ der Kovarianzmatrix. Um eine geeignete Belegung zu ermitteln, wurden zunächst drei vortrainierte Netze und zwei Maschinentypen mit SNR= 0dB und ID= 0 zufällig ausgewählt. Anschließend wurde die durchschnittliche AUC mit einer unterschiedlichen Anzahl von Normalverteilungen $\{1, 4, 8, \ldots 28\}$ mit vollbesetzten und diagonalen Kovarianzmatrizen berechnet, vgl. Abbildung 4.2.

Während die mittels MusiCNN extrahierten Repräsentationen in Bezug auf die Anzahl der Normalverteilungen und den Kovarianztyp stabil erscheinen, verschlechtern sich die Ergebnisse für andere Netze bei steigender Anzahl von Normalverteilungen und vollbelegter Kovarianzmatrix. In diesem Fall ist die Überanpassung des GMMs festzustellen. Die Normalverteilungen decken den gesamten Merkmalsraum ab und nicht nur die Bereiche, die von den Normalfalldaten belegt werden. Durch die Verwendung einer diagonalen Kovarianzmatrix lässt sich dieser Effekt vermeiden.
Alle GMMs werden mit 20 multivariaten Normalverteilungen und diagonalen Kovarianzmatrizen trainiert.

4.2.3.3 Diskussion und Interpretation der Evaluation

In diesem Abschnitt werden zunächst die Ergebnisse unter Berücksichtigung der Domäne erörtert. Erkenntnisse und Schlussfolgerungen, die auf alle Domänen zutreffen, werden darauffolgend besprochen. Schließlich wird der Ansatz mit DRINK (Abschnitt 3.3) verglichen und kurz auf die Grenzen der Ergebnisauswertung eingegangen.

SNR		-6dB				0dB				6dB			
Maschine	ID	**AE**	**Umweltg.**	**Bilder**	**Musik**	**AE**	**Umweltg.**	**Bilder**	**Musik**	**AE**	**Umweltg.**	**Bilder**	**Musik**
Lüfter	0	53.0±1.6	58.7±1.2	57.3±1.3	61.4±1.0	58.0±3.5	67.0±0.8	62.7±0.8	71.6±0.4	74.5±1.7	83.7±1.3	74.1±0.8	88.5±0.6
	2	68.7±1.6	70.9±1.1	58.5±1.7	66.6±1.7	85.9±1.5	85.1±0.6	69.7±0.9	82.9±0.9	93.0±1.3	92.8±0.5	81.4±0.5	96.4±0.5
	4	56.8±1.2	56.3±1.8	52.0±1.1	52.8±1.5	76.5±2.1	76.5±0.7	65.9±1.4	79.6±1.4	91.5±1.2	93.5±0.6	85.9±1.1	98.9±0.7
	6	78.9±2.1	87.4±0.6	87.8±1.0	94.8±1.0	93.9±0.8	94.7±0.2	97.7±0.1	99.9±0.1	98.7±0.3	98.4±0.2	99.4±0.1	100.0±0
	⌀	64.4±10.4	68.3±12.7	63.9±14.4	**68.9±16.2**	78.6±13.7	80.8±10.5	74.0±14.3	**83.5±10.6**	89.4±9.2	92.1±5.5	85.2±9.5	**95.9±4.6**
Pumpe	0	71.4±2.8	78.6±1.5	75.8±2.1	84.0±1.5	72.5±1.7	87.9±1.2	85.1±1.1	85.4±0.4	87.1±1.0	96.0±0.5	94.5±0.6	94.1±0.4
	2	53.2±3.0	55.4±1.1	68.1±2.1	62.3±1.3	55.1±2.4	65.9±2.5	90.4±1.3	77.8±2.0	60.1±1.5	74.3±1.2	97.6±0.4	81.6±0.9
	4	93.0±2.3	96.1±1.3	86.5±1.1	76.2±2.4	99.6±0.6	99.5±0.3	96.7±1.1	88.3±1.6	100.0±0	100.0±0.	99.8±0.1	98.9±1.0
	6	71.8±3.0	61.6±3.2	58.7±3.8	67.2±2.6	88.1±2.6	83.1±1.8	79.9±2.4	86.0±0.7	97.8±1.0	97.9±0.3	96.4±1.6	98.7±0.1
	⌀	72.4±14.5	**72.9±16.4**	72.3±10.7	72.4±8.8	78.8±17.0	84.1±12.5	**88.0±6.5**	84.4±4.2	86.2±16.1	92.0±10.6	**97.1±2.2**	93.3±7.2
Gleitsch.	0	91.5±1.9	98.5±0.3	99.5±0.3	99.1±0.3	96.4±0.8	99.8±0.1	99.9±0.1	99.9±0.1	99.2±0.3	100.0±0	100.0±0	100.0±0
	2	77.9±1.6	83.0±0.9	91.7±1.0	83.2±1.0	86.5±0.6	93.1±0.8	97.0±0.5	92.7±0.6	94.4±0.8	99.0±0.3	99.5±0.2	98.9±0.2
	4	71.0±2.7	82.0±1.4	70.6±2.3	79.2±1.7	88.9±3.0	95.6±0.7	84.0±1.0	94.4±0.8	95.6±1.8	98.8±0.3	92.4±0.5	98.6±0.4
	6	55.3±2.8	66.8±2.4	56.8±3.8	69.8±1.8	61.8±3.1	87.3±2.8	63.3±3.1	86.7±2.2	71.7±4.6	95.9±1.3	74.9±4.4	96.4±0.9
	⌀	73.9±13.4	82.6±11.6	79.7±17.5	**82.8±10.9**	83.4±13.3	**93.9±4.8**	86.1±14.9	93.5±5.0	90.2±11.3	98.4±1.7	91.7±10.6	**98.5±1.4**
Ventil	0	50.3±4.7	61.0±1.7	73.4±1.7	70.3±2.2	51.4±2.4	75.4±3.6	88.6±1.7	78.1±2.9	60.5±8.7	74.2±0.7	90.5±1.7	75.2±1.2
	2	62.0±3.5	74.9±1.9	63.9±3.0	70.3±2.4	72.2±2.9	82.0±2.1	70.8±1.7	80.2±1.6	75.0±6.0	85.0±1.0	72.6±1.5	86.3±0.8
	4	61.5±3.0	64.8±3.3	64.2±2.0	68.2±1.7	65.6±4.2	74.6±1.9	70.8±2.7	79.6±2.6	66.9±3.4	82.4±1.8	85.4±2.5	84.2±0.8
	6	51.2±2.9	55.8±2.6	59.4±3.3	58.7±2.9	57.5±2.0	57.8±3.6	55.9±1.7	60.7±2.6	66.3±5.0	72.0±0.8	66.0±2.1	74.2±1.1
	⌀	56.2±6.6	64.2±7.5	65.2±5.8	**66.9±5.4**	61.7±8.5	72.5±9.6	71.5±12.0	**74.7±8.6**	67.2±7.8	78.4±5.7	78.6±10.2	**80.0±5.5**

Tabelle 4.3: Durchschnittliche AUCs für alle Experimente ± eine Standardabweichung.

Maschine	ID	Transfer	DRINK
Lüfter	0	**71.6**±0.4	50.9±1.7
	2	85.1±0.6	**94.6**±1.0
	4	79.6±1.4	**87.0**±1.1
	6	**99.9**±0.1	89.7±1.6
Pumpe	0	**87.9**±1.2	63.1±1.6
	2	**90.4**±1.3	63.53±2.5
	4	**99.5**±0.3	**99.5**±0.2
	6	86.0±0.7	**93.8**±1.7
Gleitschiene	0	**99.9**±0.1	98.14±0.4
	2	**97.0**±0.5	92.13±1.0
	4	95.6±0.7	**97.33**±1.2
	6	**87.3**±2.8	75.10±2.8
Ventil	0	88.6±1.7	**94.33**±2.3
	2	82.0±2.1	**91.50**±2.2
	4	79.6±2.6	**92.47**±4.4
	6	60.7±2.6	**89.70**±6.4

Tabelle 4.4: Vergleich des vorgestellten Ansatzes (Transfer) mit DRINK aus Abschnitt 3.3. Jede Zelle stellt die durchschnittliche AUC ± eine Standardabweichung dar. Es werden jeweils immer die besten Ergebnisse (bezüglich Trainingsdomäne und Hyperparameter) miteinander verglichen.

Bilder: Mit Aufnahme der Gleitschienen übertrifft DenseNet121 das ResNet34 bei allen Maschinentypen in Bezug auf den mittlel- und den Medianwert der AUC. Die auffälligsten Leistungsunterschiede sind bei Pumpen und Ventilen zu beobachten. Eine mögliche Erklärung für die Überlegenheit von DenseNet121 ist, dass es ResNet34 auch in Bezug auf die Klassifizierungsgenauigkeit auf ImageNet übertrifft. Diese Überlegenheit überträgt sich durch die vollständigere, nuanciertere und differenziertere Extraktion von Merkmalen auch auf die akustische Anomalieerkennung.

Musik: Aus Abbildung 4.3 lässt sich entnehmen, dass L3music bei Gleitschienen und Ventilen deutlich bessere Ergebnisse erzielt als MusiCNN. Andererseits übertrifft MusiCNN L3music bei Lüftern und Pumpen.
Während Lüfter und Pumpen ein stationäres Klangmuster besitzen, weisen Schieber und Ventile ein nicht-stationäres Muster auf. Daraus wird geschlossen, dass die rechteckigen horizontalen Filter (Zeitdimension) von MusiCNN besonders gut geeignet sind, um Merkmale zu extrahieren, die den Normalbetrieb stationärer Geräusche beschreiben, da diese nahezu konstant sind und sich im Laufe der Zeit nur geringfügig verändern. Anomalien zeichnen sich dann durch das Fehlen bestimmter Merkmale aus, z.B. weil die Maschine zufällig den Betrieb einstellt, plötzlich ungewöhnliche Geräusche von sich gibt oder sich die Tonhöhe durch Beschädigungen verändert. Für Gleitschienen und Ventile könnte MusiCNN's Trainingsziel (Musik-Tagging) im Vergleich zu der sehr generellen audiovisuellen Korrespondenzaufgabe von L3music zu restriktiv sein.

Umweltgeräusche: Abbildung 4.4 zeigt die Überlegenheit von L3env gegenüber VGGish bei der Erkennung von anomalen Geräuschen von Ventilen und Gleitschienen. Bei Lüftern und Pumpen liegen beide jedoch gleichauf.
Genau wie in Abschnitt 4.2.3.3 sind die deutlichsten Leistungsunterschiede bei nicht-stationären Maschinengeräuschen zu beobachten. Hierbei wurde festgestellt, dass die VGGish Repräsentationen bei Gleitschienen und Ventilen hochgradig redundant sind. Genauer bedeutet das, dass die ursprünglichen Repräsentationen mit der Hauptkomponentenanalyse auf eine kleine Anzahl von Merkmalen reduziert werden. VGGish extrahiert also sehr ähnliche Merkmale für alle Aufnahmen, was die Erstellung eines aussagekräftigen Modells des Normalfalls erheblich erschwert.
Ferner wurden in einer früheren Arbeit [Cra+19] L3 Repräsentationen für die Erkennung von akustischen Ereignissen verwendet, wobei diese im Vergleich zu VGGish bessere Ergebnisse erzielten. Diese Ergebnisse werden hier im Kontext der akustischen Anomalieerkennung nochmals bestätigt. Darüber hinaus ist davon auszugehen, dass die selbstüberwachte audiovisuelle Korrespondenzaufgabe einen umfassenderen Satz von Merkmalen extrahiert. Im Vergleich zu dem sehr eng gefassten Klassifizierungsziel von VGGish werden für die Korrespondenzaufgabe expressivere und universellere Merkmale

benötigt.

Allgemeine Beobachtungen: Im Allgemeinen verringert die Zunahme von Hintergrundgeräuschen die Leistung aller Repräsentationen erheblich. Darüber hinaus sind alle Repräsentationen in gleicher Weise von Hintergrundgeräuschen betroffen, d. h. Hintergrundgeräusche ändern nichts an der Rangfolge.
Lüfter sind der Maschinentyp, der am meisten unter Hintergrundgeräuschen leidet. Der Grund dafür ist, dass Lüftergeräusche und Hintergrundgeräusche ein ähnliches Klangbild besitzen, wodurch sie schwerer zu differenzieren sind.
Am schwersten lassen sich anomale Geräusche bei Aufnahmen von Ventilatoren (stark nicht-stationär) erkennen, am leichtesten bei Aufnahmen von Gleitschienen (deutlich sichtbare Anomalien in den Mel-Spektrogrammen).
Gemittelt über den Maschinenidentifikator (Tabelle 4.3) schneiden alle Ansätze besser als die AE-Baseline ab. Somit bestätigt sich erneut, dass der Transfer von Wissen aus vortrainierten Netzen eine sinnvolle Alternative darstellt.
Musikbasierte Repräsentationen erreichen die besten Ergebnisse mit 8/12 Bestleistungen. Sowohl bild- als auch umweltgeräuschbasierte Repräsentationen erreichen jeweils 2/12 der Bestleistungen.
Aufgrund der starken strukturellen Ähnlichkeit von Umwelt- und Maschinengeräuschen, hätte vermutet werden können, dass auf Umgebungsgeräuschen basierende Repräsentationen die besten Resultate liefern. Die vorliegenden Ergebnisse stellen diese Annahme infrage.
Womöglich bieten Musikstücke ein reichhaltigeres und vielfältigeres Trainingssignal als Umgebungsgeräusche.

Vergleich mit DRINK: Zunächst fällt auf, dass der vorgestellte Ansatz der einzige ist dem es gelingt, bei Aufnahmen des Lüfters mit ID 0 eine AUC zu erreichen, die deutlich über 50 liegt. Keiner anderen Methode, die nicht auf dem Transfer von Repräsentationen beruht, gelingt dies. Hier ist zu beobachten, dass DRINK selbst anormale Aufnahmen gut rekonstruiert. DRINK generalisiert in diesem Fall zu gut, während der Transferansatz auf sein Vorwissen zurückgreifen kann, um auch zwischen subtileren Anomalien, wie z.B. kurzen Ausschlägen im Mel-Spektrogramm, zu unterscheiden. Für die restlichen Lüfter IDs ergibt sich kein klares Bild.
Allgemein ist jedoch eine leichte Überlegenheit des Transferansatzes gegenüber DRINK bei Aufnahmen von Lüftern, Pumpen und Gleitschienen zu beobachten. In den Fällen, in denen DRINK schlechter abschneidet, sind die Rekonstruktionsfehler selbst für Anomalien niedrig. Schlechtere Ergebnisse des Transferansatzes können zwei Gründe haben: Entweder das vortrainierte Netz verwirft wichtige Muster im Mel-Spektrogramm, die für die Anomalieerkennung wichtig wären, oder einige Muster in den Mel-Spektrogrammen des Normalfalls kommen nur selten im Datensatz vor und ihnen wird so nur

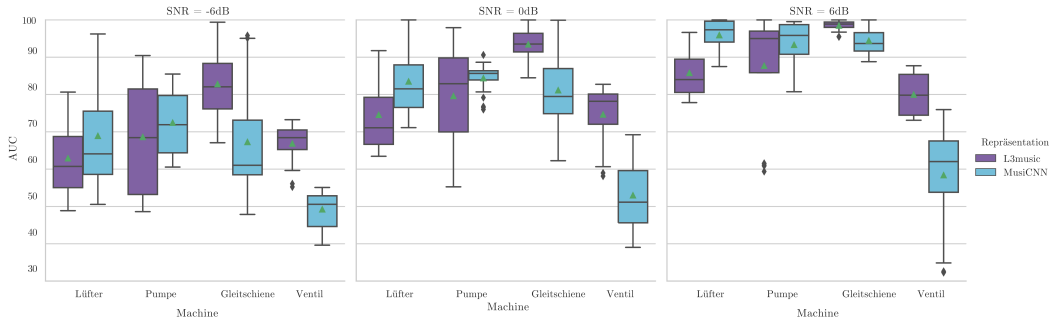


Abbildung 4.3: Vergleich der **musikbasierten** Repräsentationen.

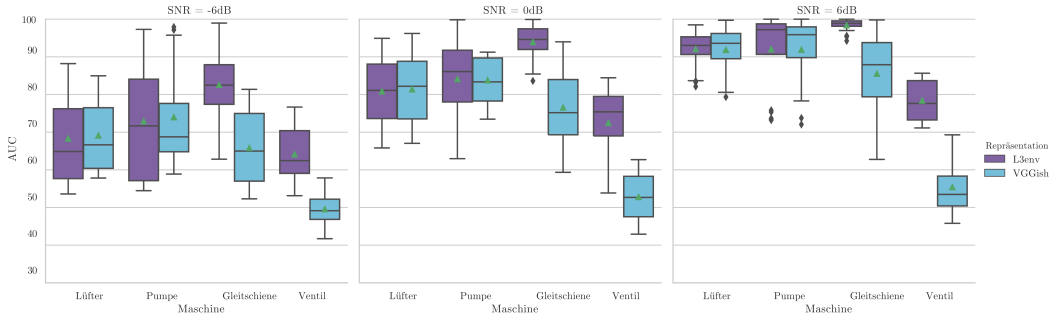


Abbildung 4.4: Vergleich der **umweltgeräuschbasierten** Repräsentationen.

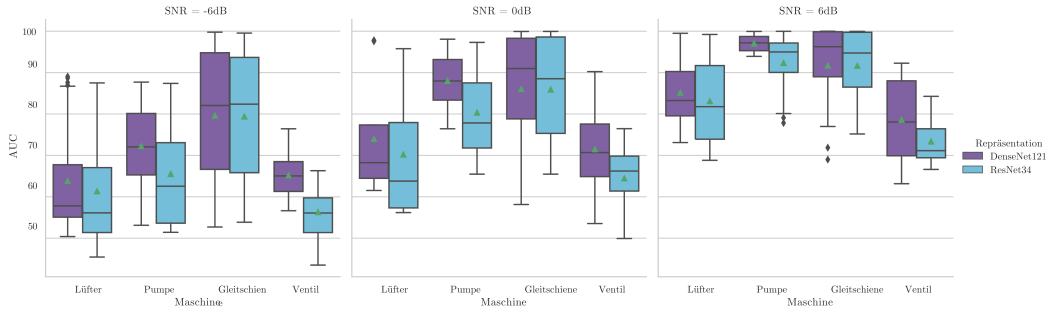


Abbildung 4.5: Vergleich der **bildbasierten** Repräsentationen.

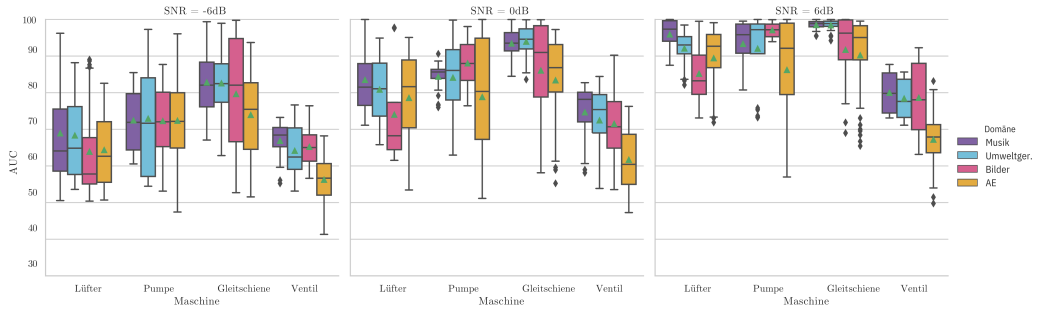


Abbildung 4.6: Vergleich der **besten** Repräsentationen (Musik: MusiCNN für Lüfter und Pumpen, L3music für Gleitschienen und Ventile; Umgebungsgeräusche: L3env; Bilder: DenseNet121).

wenig Wahrscheinlichkeitsmasse durch das GMM zugeordnet.
Bei Aufnahmen von Ventilen übertriff DRINK den Transferansatz jedoch ohne Ausnahme. Wie schon in Abschnitt 3.3.3 festgestellt, sind Kontextinformationen (umliegende Zeitrahmen) bei der Anomalieerkennung in Ventilgeräuschen besonders wichtig. Der Transferansatz berechnet seine Anomaliebewertungen pro Segment unabhängig von den umliegenden Segmenten (Gleichung 4.3) und ist folglich für stark zeitabhängige Anomalien schlechter geeignet.

Grenzen der Analyse: Die Bewertung der verschiedenen Repräsentationen stellt eine besondere Herausforderung dar. Es ist schwierig, die genauen Gründe dafür zu finden, warum eine Repräsentation besser als eine andere funktioniert. Zum einen wurden die Netze auf verschiedenen Datensätzen mit einer unterschiedlichen Anzahl an Datenpunkten trainiert, zum anderen verwenden sie alle unterschiedliche (Black-Box) Netzarchitekturen. Darüber hinaus unterscheiden sie sich in der Domäne und dem initialen Optimierungsziel. Hinzu kommt, dass der verwendete Datensatz keine Informationen darüber enthält, welche spezifischen Anomalien in einer anomalen Aufnahme vorkommen. So lässt sich nicht untersuchen, für welche Arten von Anomalien welche Repräsentation besonders gut geeignet ist. Bedauerlicherweise existiert zum aktuellen Zeitpunkt kein frei verfügbarer Datensatz zur akustischen Anomalieerkennung, der eine feingranularere Untersuchung zuließe.
In Bezug auf den Vergleich des Transferansatzes mit DRINK muss festgehalten werden, dass jeweils die besten Konfigurationen miteinander verglichen wurden. In der Praxis könnte es aufgrund der geringen Verfügbarkeit von anomalen Daten schwierig sein, die besten Konfigurationen zu finden.

4.2.4 Fazit und Ansatzpunkte zur Reduzierung der Abhängigkeit von der Domäne

Im Rahmen dieser Arbeit wurde die Wirksamkeit des Wissenstransfers von vortrainierten neuronalen Netzen für die Erkennung anormaler Geräusche unter Verwendung des Paradigmas der Merkmalsextraktion untersucht. Der Ansatz wurde mit Repräsentationen aus den Domänen Umgebungsgeräusche, Bilder und Musik evaluiert und ermöglicht schnelles Experimentieren und umgehende Ergebnisse, da nur einfache Anomalieerkennungsmodelle trainiert werden müssen. Es konnte gezeigt werden, dass fast alle Repräsentationen eine konkurrenzfähige Leistung erbringen, wobei die musikbasierten Repräsentationen besonders gut zur akustischen Anomalieerkennung geeignet sind. So konnte nachgewiesen werden, dass die Diskrepanz zwischen den Domänen weniger ins Gewicht fällt, als man anfangs hätte vermuten können. Dadurch erhöht sich die Flexibilität bei der Wahl eines passenden vortrainierten Netzes.
Die wichtigste Erkenntnis dieser Arbeit ist, dass ein relativ einfacher Versuchsaufbau, der auf Transferlernen basiert, konkurrenzfähig ist. Darüber hin-

aus entfällt die Notwendigkeit, ein völlig neues Modell zu entwickeln. Zukünftige Ansätze sollten sich folglich mit bereits vortrainierten Netzen vergleichen. Zusätzlich bietet diese Arbeit eine Orientierungshilfe bei der Auswahl von Merkmalsextraktoren für zukünftige Forschungsarbeiten.
Die Kombination von Repräsentationen aus den unterschiedlichsten Domänen zu einer gemeinsamen Repräsentation könnte die Abhängigkeit von der Wahl der richtigen Domäne reduzieren. Sind einige Anomalien bereits zum Trainingszeitpunkt verfügbar, ließe sich mittels Supervised Learning ein FFNN trainieren, welches aus den einzelnen Repräsentationen die geeigneten Merkmale extrahiert und kombiniert.

4.3 Kapitelzusammenfassung

Dieses Kapitel widmete sich dem Transfer des Wissens von vortrainierten neuronalen Netzen zur Erkennung von Anomalien in Aufnahmen von Industriemaschinen. Durch dieses Vorgehen muss keine neue Netzarchitektur entworfen werden und nicht von Grund auf neu trainiert werden.
Abschnitt 4.1 untersuchte dazu zunächst die Auswirkungen der Verwendungen von zur Bildklassifizierung trainierten CNNs. Es wurde gezeigt, dass die so extrahierten Repräsentationen unter Verwendung von One-Class Support Vector Machines und Gaussian Mixture Models bessere Ergebnisse erzielen als Convolutional Autoencoder.
Anschließend wurden in Abschnitt 4.2 vortrainierte neuronale Netze aus weiteren Domänen berücksichtigt. Hier konnte ein klarer Vorteil für musikbasierte Repräsentationen festgestellt werden. Folglich ist die Diskrepanz zwischen den Domänen weniger relevant als ursprünglich vermutet. Ferner konnte eine leichte Überlegenheit musikbasierter Repräsentationen in Bezug auf den in Abschnitt 3.3 vorgestellten Ansatz DRINK festgestellt werden.

5 Repräsentationslernen in weiteren Anwendungsfeldern

Die vorangegangenen Kapitel beschäftigten sich ausschließlich mit dem maschinellen Lernen auf akustischen Daten. Im Gegensatz dazu wird in diesem Kapitel das Spektrum der Anwendungen erweitert. Der Fokus liegt dabei auf der Frage, wie das Repräsentationslernen im Multi-Agent Reinforcement Learning und dem Mannschaftssport eingesetzt werden kann.
Im Multi-Agent Reinforcement Learning lernen die beteiligten Agenten geeignete Handlungsstrategien (Policies) durch die Interaktion mit der Umgebung mittels Versuch und Irrtum. Abschnitt 5.1 stellt einen neuen Ansatz zur Ermöglichung von effizienter und diskreter Kommunikation zwischen den Agenten vor. Hierbei werden die internen Repräsentationen der Agenten fortlaufend durch ein Clusteringverfahren diskretisiert und die Clusterindizes als Nachrichten versendet.
Abschnitt 5.2 ist ebenfalls dem Forschungsfeld des Reinforcement Learnings zuzuordnen. Hier wird die Anomalieerkennung aus den vorangegangenen Kapiteln wieder aufgegriffen und diskutiert, wie diese auf das Reinforcement Learning übertragen werden kann. Der Abschnitt unterscheidet sich zu den in der vorliegenden Arbeit vorgestellten Ansätzen darin, dass er keine konkreten Lösungsansätze und Experimente beinhaltet. Der Schwerpunkt liegt auf der Eingrenzung der Problemstellung, da diese in der Literatur aktuell nicht ausreichend präzise formuliert ist.
Abschließend wird STEVE vorgestellt, ein Algorithmus zum automatischen Erlernen von Repräsentationen von Fußballteams. Der Ansatz benötigt lediglich frei verfügbare Spielergebnisse aus verschiedenen Fußballligen und Wettbewerben. Somit wird die Abhängigkeit von teuren Daten von Sportanalyseunternehmen und Fachexperten verringert.
Abschnitt 5.1 stellt insbesondere Inhalte und Ergebnisse vor, die noch nicht veröffentlicht wurden. Abschnitt 5.2 und Abschnitt 5.3 basieren auf den Publikationen [Mül+22] bzw. [Mül+20].
Am Ende dieses Kapitels fasst Abschnitt 5.4 die wichtigsten Erkenntnisse kurz zusammen. Eine detailliertere Zusammenfassung der vorgestellten Ansätze sowie mögliche Ansatzpunkte für zukünftige Arbeiten befindet sich jeweils am Ende des entsprechenden Abschnitts.

5.1 Agentenkommunikation durch Clustering der Repräsentationen

Die Lösung von Problemen wie dem autonomen Fahren [SSS16] oder der sicheren Koordination von Robotern in Lagerhäusern [SS20] erfordert in den meisten Fällen die Kooperation der beteiligten Agenten. Um gemeinsam Aufgaben und Probleme zu lösen, müssen die Agenten lernen, sich selbstständig zu koordinieren und Ressourcen zu teilen. Ähnlich wie beim Menschen wird die Kooperation durch die Kommunikation erleichtert. Autonome Agenten sollten folglich so modelliert und trainiert werden, dass sie in der Lage sind, eigenständig zu handeln und gleichzeitig mit anderen Agenten zu kommunizieren. Dadurch können Agenten beispielsweise Informationen über ihre aktuellen Wahrnehmungen, ihr geplantes Handeln oder ihren internen Zustand austauschen, was ihnen die Zusammenarbeit bei Aufgaben und Problemen ermöglicht, die allein und ohne Kommunikation schwieriger oder unmöglich zu lösen sind.
Im *Multi-Agent Reinforcement Learning* (MARL) interagieren mehrere Agenten mit einer gemeinsamen Umgebung. Im einfachsten Fall wird jeder Agent als unabhängig lernende Einheit modelliert, welche die anderen Agenten lediglich wahrnimmt (*Independent Learning*). In diesem Szenario kann jedoch höchstens implizite Kommunikation entstehen, wie sie zum Beispiel bei Bienen, die ihre Artgenossen durch spezielle Bewegungsabläufe über die Position einer gefundenen Futterquelle informieren [Von92], vorkommt. Im Kontext des MARLs wurde ferner bereits festgestellt, dass die Kooperation durch den Austausch von Informationen, Erfahrungen und dem erlernten Wissen der einzelnen Agenten verbessert werden kann [Min].
Der vorherrschende Ansatz dafür besteht darin, den Agenten während des Trainings zusätzlich zum eigenen Zustand weitere Informationen wie z.B. den Gesamtzustand der Umgebung zum aktuellen Zeitpunkt, zu übergeben. Meist ist es wünschenswert, die Agenten nach dem Training dezentral ausführen zu können, d.h. jeder Agent soll nur anhand ihm lokal zur Verfügung stehender Informationen entscheiden können, wie er handelt. Folglich dienen die zusätzlichen Informationen zum Trainingszeitpunkt nur dazu, das Training zu beschleunigen und ausgefeiltere, implizite Koordinations- und Kommunikationsstrategien zu entwickeln. Dieses Paradigma wird *Centralized Learning and Decentralized Execution* genannt.
Ein noch radikalerer Ansatz besteht darin, alle Agenten als einen einzigen RL Agenten aufzufassen, der alle lokalen Beobachtungen der einzelnen Agenten erhält und den gemeinsamen Aktionsraum als kartesisches Produkt der individuellen Aktionsräume auffasst. Da der gemeinsame Aktionsraum mit der Anzahl der Agenten exponentiell anwächst und die Agenten nicht dezentral ausgeführt werden können, kommt dieser Ansatz in praktischen Problemen selten zum Einsatz.
Damit Agenten sich nicht nur implizit miteinander austauschen können, führen neuere Ansätze [Foe+16; SSF16; Low+17; Das+19; Lin+21; WS22] einen expli-

ziten Kommunikationskanal zwischen den Agenten ein. Dies stellt ein beträchtliches Explorationsproblem dar, da die Wahrscheinlichkeit, dass ein einheitliches Kommunikationsprotokoll gefunden wird, äußerst gering ist [Foe+16]. Des Weiteren tragen die initial noch zufälligen Nachrichten zu der bereits hohen Varianz des MARLs bei und erschweren das Training zusätzlich [Low+17; Ecc+19]. Viele Arbeiten greifen daher auf differenzierbare Kommunikation zurück [SSF16; Low+17; CLF18; MA18; Ecc+19; WS22], bei der die Agenten die Kommunikationsstrategien der anderen Agenten direkt durch Gradienten optimieren können.
Keines der angesprochenen Szenarien ist hinreichend realistisch. Weder können Menschen die Neuronen des Gegenübers anpassen, noch ist es immer möglich während eines Lernprozesses zusätzliche Informationen über den Gesamtzustand der Umgebung zu sammeln.
Darüber hinaus ist es im MARL üblich, dasselbe neuronale Netz für alle Agenten zu verwenden (*Parameter Sharing*). So kann das Netz von der gesamten Erfahrung aller Agenten profitieren und das Training beschleunigen.
Auch diese Annahme lässt sich nicht darauf übertragen, wie Menschen lernen. Überdies ist anzunehmen, dass das gemeinsame Training von Agenten in der echten Welt (ohne Simulationsumgebung) aus Gründen der Skalierbarkeit ohne zentrale Steuerungseinheit auskommen sollte.
Aus den genannten Gründen wird in dieser Arbeit angenommen, dass jeder Agent ein eigenes neuronales Netz besitzt, ein nicht-differenzierbarer Kommunikationskanal besteht, keine zentrale Steuerungseinheit das Training überwacht und die Agenten zusätzliche Informationen ausschließlich über den Kommunikationskanal erhalten können.
Ansatt den Nachrichtenaustausch durch nicht-differenzierbare Optimierungsverfahren zu lernen, wird hier direkt die interne Repräsentation eines Agenten zur Erstellung einer Nachricht verwendet. Die interne Repräsentation ist dabei durch die Aktivierungen der letzten versteckten Schicht des Agenten gegeben. Durch das Clustering der internen Repräsentationen kann anschließend eine diskrete Nachricht in Form des Indizes des zugewiesenen Clusters erstellt werden. Die diskrete Kommunikation entspricht eher der Kommunikation in der Natur und kann einen größeren praktischen Nutzen bieten (z. B. Kommunikationskanäle mit geringer Bandbreite zwischen Robotern [Lin+21]). Zu jedem Zeitschritt erhält jeder Agent jeweils die Indizes (Nachrichten) aller anderen Agenten und berechnet auf Basis der erhaltenen Indizes und seines lokalen Zustandes die nächste auszuführende Aktion.
Es kann gezeigt werden, dass dieses Verfahren im Vergleich zu keiner Kommunikation bessere Kooperation ermöglicht und mit der Leistung des Austauschs der gesamten kontinuierlichen Repräsentationen konkurriert.

5.1.1 Verwandte Arbeiten und Grundlagen

In diesem Abschnitt werden zunächst verwandte Arbeiten besproche und anschließend die grundlegenden Prinzipien vorgestellt, auf denen diese Arbeit fußt. Dabei bilden Markov-Spiele den zu Grunde liegenden Formalismus für das MARL und K-Means das Clusteringverfahren, welches zur Diskretisierung der Nachrichten verwendet wird.

5.1.1.1 Verwandte Arbeiten zur Kommunikation in Multiagentensystemen

Die Koordination von voneinander unabhängig trainierten Agenten mit der Möglichkeit zu kommunizieren wurde bereits von Jaques et al. [Jaq+19] untersucht. In jedem Zeitschritt simuliert ein Agent durch kontrafaktisches Schließen alternative Aktionen, die er hätte ausführen können, und berechnet deren Auswirkungen auf das Verhalten anderer Agenten. Aktionen, die zu größeren Veränderungen im Verhalten anderer Agenten führen, gelten als einflussreich und werden belohnt. Jedoch setzt diese Methode voraus, dass ein Agent Zugang zu den *Policies* (Strategien) der anderen Agenten hat. Der vorliegende Ansatz erfordert weder die kontrafaktische Simulation von Aktionen noch benötigt er Zugang zu den gesamten Policies aller anderen Agenten.
Die gleiche Problemstellung wird auch von Eccles et al. [Ecc+19] untersucht. Hier werden die Agenten explizit regularisiert, um aussagekräftige Nachrichten zu erstellen (*Positive Signaling*) sowie auf eingehende Nachrichten zu reagieren (*Positive Listening*). Eine solche Regularisierung ist im vorliegenden Ansatz nicht erforderlich, denn die Art der Nachrichten ist bereits extern festgelegt. Die Experimente belegen, dass die Agenten auch ohne zusätzlichen Anreiz auf die Nachrichten reagieren.
Am engsten verwandt ist der in dieser Arbeit vorgestellte Ansatz mit AEcomm [Lin+21]. Hier tauschen Agenten ebenfalls diskrete Indizes aus. Im Gegensatz zu der vorliegenden Arbeit nutzt AEcomm zur Generierung von Nachrichten jedoch einen zusätzlichen Autoencoder. Jeder Agent trainiert dabei seinen eigenen Autoencoder, welcher den lokalen Zustand des Agenten auf eine geringe Anzahl von Aktivierungen reduziert, mit Hilfe des *Straight-Through Estimator* [BLC13] diskretisiert und anschließend rekonstruiert. Hier ist also das Training eines zusätzlichen Netzwerks mit einer zusätzlichen Trainingsaufgabe nötig. Darüber hinaus stellten die Autoren fest, dass AEcomm nur geringfügige Vorteile gegenüber der Verwendung der gesamten internen Repräsentation des Agenten bietet. Auf Basis dieser Erkenntnis wurde der Ansatz in dieser Arbeit entwickelt. Er diskretisiert die internen Repräsentationen der Agenten durch iteratives Clustering während des Trainings und versendet die Clusterindizes. Dadurch entfällt das Training eines zusätzlichen neuronalen Netzwerks.

5.1.1.2 Markov-Spiele

Diese Arbeit folgt der üblichen Modellierung des MARLs als partiell beobachtbares Markov-Spiel (*Partially Observable Markov Game*, POMG) [Sha53; Lit94; HBZ04] mit N Agenten. In jedem Zeitschritt erhält jeder Agent eine Teilbeobachtung des tatsächlichen Gesamtzustands. Diese Beobachtungen werden wiederum verwendet, um geeignete Policies zu erlernen, welche den individuellen Reward maximieren.
Formal ist ein POMG gegeben durch das Tupel $(\mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{O}, \Omega, \mathcal{P}, \mathcal{R}, \gamma)$ mit $\mathcal{N} = \{1, \dots, N\}$. Ferner beschreibt $\mathcal{S}$ die Zustandsmenge, $\mathcal{A} = \{\mathcal{A}^{(1)}, \dots, \mathcal{A}^{(N)}\}$ die Aktionsmenge und $\mathcal{O} = \{\mathcal{O}^{(1)}, \dots, \mathcal{O}^{(N)}\}$ den Beobachtungsraum.
Zum Zeitpunkt t beobachtet Agent $k \in \mathcal{N}$ einen lokalen Teilausschnitt $o_t^{(k)} \in \mathcal{O}^{(k)}$ des zugrundeliegenden Gesamtzustands $s_t \in \mathcal{S}$ und wählt daraufhin die nächste auszuführende Aktion $a_t^{(k)} \in \mathcal{A}^{(k)}$. Die stochastischen Beobachtungen der Agenten hängen dabei von den ausgeführten Aktionen aller Agenten sowie dem Gesamtzustand ab $\Omega(o_t^{(1)}, \dots o_t^{(N)} | s_t, a_{t-1}^{(1)}, \dots, a_{t-1}^{(N)})$. Auf gleiche Weise hängt der stochastische Gesamtzustand von dem vorangegangenen Zustand und den gewählten Aktionen aller Agenten ab $\mathcal{P}(s_t | s_{t-1}, a_{t-1}^{(1)}, \dots, a_{t-1}^{(N)})$. Schließlich erhält jeder Agent k einen reward gemäß der individuellen Rewardfuktion $\mathcal{R}^{(k)} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R},\ \mathcal{R} = \{R^{(1)}, \dots, R^{(N)}\}$. Sei dabei $r_t^{(k)}$ der Reward, den Agent k zum Zeitpunkt t erhält, und die Policy des Agenten $\pi^{(k)}(a_t^{(k)} | o_t^{(k)})$ eine Verteilung über die zu wählende Aktion in Abhängigkeit der erhaltenen lokalen Beobachtung.
Das Ziel ist es, Policies $\pi = (\pi^{(1)}, ..., \pi^{(N)})$ zu finden, sodass der mit γ diskontierte Reward[1] über den Zeithorizont $T \in \mathbb{N}$ einer Episode[2] für alle Agenten k

$$G^{(k)} = \sum_{t=0}^{T} \gamma^t r_t^{(k)} \tag{5.1}$$

im Bezug auf die anderen Policies in π maximiert wird:

$$\forall_k : \pi^{(k)} \in \arg\max_{\tilde{\pi}^{(k)}} \mathbb{E}[G^{(k)} | \tilde{\pi}^{(k)}, \pi^{(-k)}] \tag{5.2}$$

wobei $\pi^{(-k)} = \pi \setminus \{\pi^{(k)}\}$. Um die Möglichkeit der Kommunikation in einem POMG explizit zu berücksichten [Lin+21], kann das POMG um die Menge von Nachrichtenräumen $\mathcal{M} = \{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}\}$ erweitert werden. Jeder Agent erhält nun zum Zeitpunkt t Zugriff auf alle Nachrichten aus dem vorangegangenen Zeitschritt $m_t = \{m_{t-1}^{(k)} \in \mathcal{M}^{(k)} \mid 1 \le k \le N\}$. Die Policy verarbeitet nun

[1] Die Diskontierung macht eine unendliche Summe endlich (wenn $T = \infty$). Ferner wird mit kleineren Werten für γ der unmittelbar erreichbare Reward bevorzugt.

[2] Eine Abfolge von Beobachtungen, Aktionen und Rewards, z.B. das Durchspielen eines Levels eines Videospiels.

ebenfalls die erhaltenen Nachrichten und berechnet zusätzlich zu der Verteilung über die nächste Aktion eine neue Nachricht $\pi^{(k)}(a_t^{(k)}, m_t^{(k)} | o_t^{(k)}, m_{t-1}^{(k)})$

5.1.1.3 K-Means

Clustering ist die gängigste Ausprägung des Unsupervised Learnings. Hierbei werden Daten auf der Grundlage ihrer Ähnlichkeit in Gruppen unterteilt, ohne dass diesen zuvor Labels zugewiesen werden müssen. Ziel ist es, eine Gruppierung zu finden, bei der die Ähnlichkeit zwischen einzelnen Objekten einer Gruppe größer ist als zwischen Objekten in anderen Gruppen. Um die Ähnlichkeit zu bestimmen, werden Distanz- oder Ähnlichkeitsmaße wie die euklidische Distanz bzw. die Kosinus-Ähnlichkeit verwendet.
K-Means [Llo82] stellt dabei eines der einfachsten und gleichzeitig am weitesten verbreitetes Clusteringverfahren dar und wird in dieser Arbeit zur Diskretisierung der Nachrichten verwendet.
Sei $D = \{x_i \in \mathbb{R}^D | 2 \leq i \leq n\}$ ein Datensatz von n Vektoren mit D Dimensionen und $\mu = \{\mu_i \in \mathbb{R}^D | 2 \leq i \leq k\}$ eine Menge von $k \geq 2$ Zentroiden derselben Dimensionalität. Das Ziel ist es nun, D so in k Partitionen zu teilen, dass die Summe der quadrierten Abweichungen (quadratische euklidische Distanz) von den Zentroiden minimal ist:

$$\arg\min_{\mu} \sum_{i=1}^{n} ||x_i - \mu_{x_i}||_2^2 \tag{5.3}$$

wobei $\mu_{x_i} := \arg\min_{\mu_j \in \mu} ||x_i - \mu_i||_2^2$. Da die Suche nach der optimalen Lösung NP-schwer ist, wird auf approximative Algorithmen wie den Lloyds Algorithmus [Llo82] zurückgegriffen. Dabei werden die Zentroide initial zufällig gewählt und anschließend jeder Datenpunkt demjenigen Zentroiden zugeordnet, zu dem die Distanz minimal ist. Die neuen Zentroide werden daraufhin auf den Mittelwert aller dem jeweiligen Zentroid zugewiesenen Datenpunkte gesetzt. Die letzten beiden Schritte werden bis zur Konvergenz wiederholt.
Aufgrund seiner Einfachheit und weiten Verbreitung wurde der K-means-Algorithmus für zahlreiche Anwendungsfälle adaptiert und aktualisiert. Zu den wichtigsten Varianten gehören fuzzy K-Means [Dun73], K-Medians [JD88] und K-Medoids [KR08]. Für die vorliegende Arbeit am relevantesten sind jedoch die Initialisierungsmethode K-Means++ [VA06] und Mini-Batch-K-means.
K-Means++ verringert den Fehler im Vergleich zur optimalen Lösung und beschleunigt die Konvergenz durch die Vergrößerung der Streuung der initialen Zentroiden. Dabei wird der erste Zentroid zufällig auf ein $x_i \in D$ gesetzt. Danach wird jeder nachfolgende Zentroid aus $D \setminus \{x_i\}$ mit einer Wahrscheinlichkeit ausgewählt, die proportional zur Distanz zum nächstgelegenen bestehenden Zentroid des Punktes ist.
Min-Batch-K-Means wurde entwickelt, um den Speicherplatzbedarf zu reduzieren. Dabei wird in jeder Iteration nur eine Untermenge von D mit fester Größe berücksichtigt (*Mini-Batches*). Darüber hinaus ist Min-Batch-K-Means

in Szenarien nützlich, in denen der gesamte Datensatz zu einem bestimmten Zeitpunkt unbekannt ist und möglicherweise zeitabhängigen Änderungen obliegt (z. B. bei Datenströmen).

5.1.2 ClusterComm

In diesem Abschnitt wird *ClusterComm* vorgestellt, ein Ansatz zur diskreten Kommunikation in Multi-Agenten Systemen. ClusterComm ist lose inspiriert von der Art und Weise, wie Menschen im Laufe der Zeit gelernt haben, zu kommunizieren. Ihre Entwicklung begann mit einfachen Handbewegungen und der Nachahmung natürlicher Laute bis hin zu gesprochener Sprache, einem etablierten Kommunikationsprotokoll [Fit10]. Da die menschliche Kommunikation ohne Beaufsichtigung oder zentralen Kontrollmechanismus und durch die Interaktion von Individuen entstanden ist, verzichtet der vorliegende Ansatz, im Gegensatz zu den meisten in Abschnitt 5.1 vorgestellten Ansätzen, auf Parameter Sharing und differenzierbare Kommunikation. Stattdessen lernen und handeln alle Agenten unabhängig voneinander, haben aber dennoch die Möglichkeit, Nachrichten über den Kommunikationskanal auszutauschen. Mit dem Ziel, einen Kommunikationsmechanismus zu schaffen, der so wenig Annahmen wie möglich, aber so viel wie nötig verwendet, um die interne Repräsentation eines jeden Agenten (letzte versteckte Schicht) mit Hilfe des Clusteringverfahrens K-Means (Abschnitt 5.1.1.3) zu diskretisieren und die resultierenden Clusterindizes als Nachricht zu verwenden. Eine Nachricht besteht folglich aus einem einzelnen Integer (ganze Zahl). Das bedeutet, dass die Anzahl der für die Kommunikation benötigten Symbole deutlich geringer ist als bei anderen Methoden [Lin+21; Foe+16]. Das Clustering ermöglicht den Agenten, Informationen zu komprimieren, indem sie Daten auf der Grundlage von Ähnlichkeiten gruppieren. So sind sie in der Lage, aussagekräftige und dennoch kompakte Nachrichten zu generieren. Diskrete Kommunikation ist besonders schwierig, da ohne zusätzliche Maßnahmen keine Gradienten durch den Kommunikationskanal fließen können [Van+22]. Sie entspricht jedoch eher der menschlichen Kommunikation, da die Entwicklung von Lauten und Gesten hin zu Symbolen und Wörtern erfolgte, die diskret sind [Tal05].

5.1.2.1 Generelle Architektur und Kommunikationsstrategie

Zu jedem Zeitschritt t erhält jeder Agent $i \in \{1, \ldots, n\}$ zwei Eingaben: die aktuelle lokale Beobachtung $o_t^{(i)}$ und die von allen anderen Agenten gesendeten Nachrichten $m_{t-1}^{(-i)} = \{m_{t-1}^{(j)} | 1 \leq j \leq n \wedge j \neq i\}$. Die lokalen Beobachtungen $o_t^{(1)} \ldots o_t^{(n)}$ sind im Allgemeinen für jeden Agenten unterschiedlich und hängen vom aktuellen Gesamtzustand der Umgebung ab. Abbildung 5.1 stellt exemplarisch die Kommunikation zwischen Agenten für einen Zeitschritt dar. Jeder Agent i nutzt einen neuronalen *Observation Encoder* $\phi_o^{(i)}$, um eine Repräsen-

tation für $o_t^{(i)}$ zu erstellen. Der *Message Encoder* $\phi_m^{(i)}$ erhält die Konkatenation von $m_{t-1}^{(-i)}$ (als one-hot encoding) und erstellt ebenfalls eine Repräsentation der erhaltenen Nachrichten. Schließlich erhält $\phi_a^{(i)}$ die Konkatenation der Repräsentationen der Nachrichten und der lokalen Beobachtung und berechnet die Wahrscheinlichkeitsverteilung über den Aktionsraum $\mathcal{A}^{(i)}$. Da $\phi_o^{(i)}(o_t^{(i)})$ Infor-

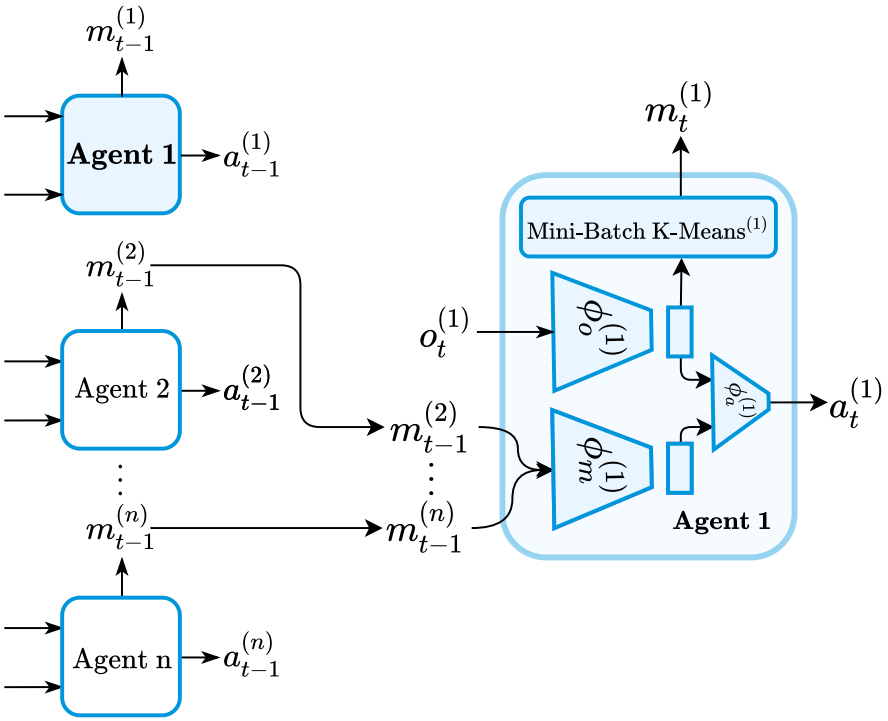


Abbildung 5.1: ClusterComm zwischen Agenten. Agent 1 erhält zum Zeitpunkt t die Beobachtung $o_t^{(1)}$ sowie die Nachrichten $m_{t-1}^{(2)}, \ldots, m_{t-1}^{(n)}$ der anderen n Agenten. Mit Hilfe seines Message Encoder $\phi_m^{(1)}$ und Observation Encoders $\phi_o^{(1)}$ werden zwei Repräsentationen erstellt, welche zur Berechnung der nächsten Aktion $a_t^{(1)}$ sowie der nächsten Nachricht $m_t^{(1)}$ herangezogen werden. Die Nachrichten werden mittels Mini-Batch K-Means$^{(1)}$ diskretisiert.

mationen über die Beobachtung und die geplante Aktion des Agenten encodiert, wird die Ausgabe mittels Mini-Batch K-Means diskretisiert. Dabei wird als Nachricht der Clusterindex des Zentroiden gewählt, der am nähesten an der Ausgabe des Observation Encoders liegt. Da die Parameter der Netze der Agenten während des Trainings kontinuierlich angepasst werden (Gradientenabstiegsverfahren), ändert sich auch der aufgespannte Merkmalsraum über den Trainingsverlauf. Die naive Anwendung von K-Means würde dazu führen, dass die Zentroide in jedem Trainingsschritt neu berechnet und die Indizes folglich neu vergeben würden. Infolgedessen würden die versendeten Nachrichten

durch die ständig wechselnden Indizes ihre Bedeutung verlieren. Der sich kontinuierlich weiterentwickelnde Merkmalsraum gleicht einem Datenstrom und folglich können durch die Verwendung von K-Mini-Batch K-Means die genannten Probleme verringert werden. Die Zentroide werden kontinuierlich mit dem Aufkommen neuer Datenpunkte angepasst, wodurch die Zuordnung seltener wechselt.
Zum Training der Agenten wird *Proximal Policy Optimization* (PPO) [Sch+17] verwendet. PPO ist einer der am weitesten verbreiteten Algorithmen des RLs und zeichnet sich dadurch aus, dass er sehr robust gegenüber der Wahl der Hyperparameter ist. Darüber hinaus wurde seine Überlegenheit im Kontext des MARLs in aktuellen Studien bestätigt [Yu+21; DVN20].

5.1.2.2 Konzeptionelle Variationen des Ansatzes

Neben dem Hauptbeitrag, dem ClusterComm-Algorithmus, werden zwei weitere Varianten vorgeschlagen.

i) **Spherical ClusterComm**: Bei dieser Variante von ClusterComm wird untersucht, welche Auswirkungen die Vektornormalisierung hat. Dazu werden die Repräsentationen mittels ScaleNorm [NS19] normalisiert und auf die $(d-1)$-dimensionale Hypersphäre mit gelerntem Radius r projiziert.

$$\text{ScaleNorm}(x; r) = r\frac{x}{\|x\|} \tag{5.4}$$

Somit wird die maximale Distanz durch r festgelegt, was verhindert, dass einzelne Dimensionen die Distanzberechnung dominieren. Darüber hinaus bewirkt die Verwendung von normalisierten Vektoren, dass das Clustering auf Basis der Kosinus-Ähnlichkeit (Winkel zwischen Vektoren) und nicht der euklidischen Distanz durchgeführt wird [Hor+12].

ii) **CentroidComm**: Obwohl ein Ziel darin besteht, die Nachrichtengröße zu reduzieren und nur diskrete Nachrichten zu verwenden, wird eine weitere Variante ohne diese Einschränkungen vorgeschlagen. Anstelle des Clusterindizes wird bei CentroidComm der näheste Zentroid direkt übermittelt. Die Nachrichten bestehen somit aus größeren, kontinuierlichen Vektoren. Somit nutzt CentroidComm während des Trainings mehr Bandbreite für die Übertragung der Nachrichten. Am Ende des Trainings können jedem Agenten die Zentroide aller anderen Agenten einmalig übermittelt und folglich zum Zeitpunkt der Ausführung der trainierten Agenten wieder Clusterindizes verschickt werden.

5.1.3 Experimentelle Untersuchung von ClusterComm

Um die Leistungsfähigkeit von ClusterComm zu evaluieren, werden in diesem Abschnitt zunächst die verwendeten Evaluationsumgebungen vorgestellt. Dar-

auf folgt die Diskussion der Ansätze, mit denen ClusterComm verglichen wird. Schließlich werden die Ergebnisse diskutiert und eingeordnet.

5.1.3.1 Evaluationsumgebungen

ClusterComm wird mit den folgenden Evaluationsumgebungen untersucht:

i) ClosedRooms: Diese einfache Umgebung besteht aus zwei Agenten, die sich in zwei getrennten, abgeschlossenen Räumen mit jeweils eigenen Zielen befinden. Der erste Agent wird Zuhörer genannt und hat zwei Ziele - eines in der linken Ecke und ein weiteres in der rechten Ecke. Der zweite Agent, Sprecher genannt, hat nur ein Ziel, das zufällig in einer der beiden Ecken platziert ist. Beide Agenten befinden sich in der Mitte ihres Raumes. Ziel ist es, dass beide Agenten ihre Ziele erreichen, allerdings müssen sich ihre Ziele in derselben Ecke befinden. Liegt das Ziel des zweiten Agenten in der linken Ecke und hat der erste Agent bereits die rechte Ecke erreicht, wird die Episode erfolglos beendet. Das bedeutet, dass der Sprecher dem Zuhörer die Seite des Ziels mitteilen muss. Diese Umgebung erfordert Kommunikation, um gelöst werden zu können. Agenten, die nicht kommunizieren, können bestenfalls eine Erfolgsquote von 0,5 erreichen, da der Zuhörer zufällig zwischen dem rechten und linken Ziel wählen muss. Jeder Agent hat fünf mögliche Aktionen: Stillstehen, Bewegung nach oben, Bewegung nach unten, Bewegung nach links, Bewegung nach rechts.

ii) Bottleneck: Diese Umgebung besteht aus zwei Räumen, die über eine einzelne Zelle (Nadelöhr) miteinander verbunden sind. Auf jeder Zelle kann sich immer nur ein Agent aufhalten. Bis zu vier Agenten werden am Anfang einer Episode gleichmäßig auf die Ecken der Räume verteilt. Das Ziel eines jeden Agenten ist es, die gegenüberliegende Ecke im anderen Raum so schnell wie möglich zu erreichen. Da sich immer nur ein Agent durch das Nadelöhr bewegen kann, müssen die Agenten ihre Bewegungen koordinieren. Die möglichen Aktionen sind dieselben wie bei ClosedRooms.

iii) RedBlueDoors [Lin+21]: Die Umgebung besteht aus zwei Agenten und zwei Türen (eine rote und eine blaue). Beide Agenten können beide Türen öffnen, wenn sie auf der Zelle neben der Tür stehen. Das Ziel ist es, zuerst die rote Tür und danach die blaue Tür zu öffnen. Wenn die blaue Tür zu einem beliebigen Zeitpunkt geöffnet wird, bevor die rote Tür geöffnet wurde, wird die Episode erfolglos beendet. Die Türen werden immer an einer zufälligen Position in der linken oder rechten Wand platziert, während die Agenten zufällig in einer leeren Zelle platziert werden. Die Agenten haben eine Teilsicht auf die Umgebung, in der sie nur einen Bereich von 5 $\times$ 5 Zellen um sich herum sehen. Dementsprechend müssen sie zunächst die Türen finden. Da beide Agenten beide Türen öffnen können, kann ein Agent die Umgebung alleine lösen. Es ist jedoch vorteilhaft, die Aufgabe

mittels Kommunikation zwischen den Agenten aufzuteilen, denn ein weiteres Ziel besteht darin, die Aufgabenstellung so schnell wie möglich zu lösen. Die Agenten haben sechs mögliche Aktionen: Stillstehen, Bewegung nach oben, Bewegung nach unten, Bewegung nach links, Bewegung nach rechts, Interagieren. Die Aktion interagieren öffnet und schließt die Tür nur, wenn der Agent auf dem Feld neben der Tür steht.

iv) Level-based Foraging [CSA20]: Diese Umgebung konzentriert sich auf die Koordination der beteiligten Agenten. Die Agenten bewegen sich und sammeln Nahrung, indem sie bei Bedarf mit anderen Agenten zusammenarbeiten. Nahrung wird zufällig in der Umgebung verteilt. Nahrung kann nur aufgesammelt werden, wenn beide Agenten auf einem der Nahrungsquelle angrenzenden Feld stehen und sich gleichzeitig dazu entscheiden die Nahrung aufzunehmen. Der Agent hat sechs mögliche Aktionen: Stillstehen, Bewegung nach oben, Bewegung nach unten, Bewegung nach links, Bewegung nach rechts, Gegenstand einsammeln. Wenn eine Aktion nicht gültig ist (z. B. gegen die Wand laufen), wird sie durch Stillstehen ersetzt.

Die Umgebungen ClosedRooms und Bottleneck wurden speziell für diese Arbeit neu entwickelt, die verbleibenden Umgebungen entstammen der Literatur. Wenn nötig, wurde die Implementierung so angepasst, dass die Agenten nur eine partielle Sicht (5×5) des Gesamtzustands sehen.
Um die Agenten zu incentivieren, die Aufgabenstellung so schnell wie möglich zu lösen, ist die Rewardfunktion im Falle von ClosedRooms, Bottleneck und RedBlueDoors wie folgt definiert:

$$r(s_t, a_t, s_{t+1}) = \begin{cases} +1.0 \text{ wenn alle Agenten ihr jeweiliges Ziel erreicht haben} \\ -(1/\text{max\_schritte}) \text{ ansonsten} \end{cases} \tag{5.5}$$

Im Fall von Level-based-Foraging wird jeder Agent proportional zur gesammelten Nahrung belohnt. Alle Umgebungen sind in Abbildung 5.2 dargestellt.

5.1.3.2 Vergleich mit weiteren Ansätzen

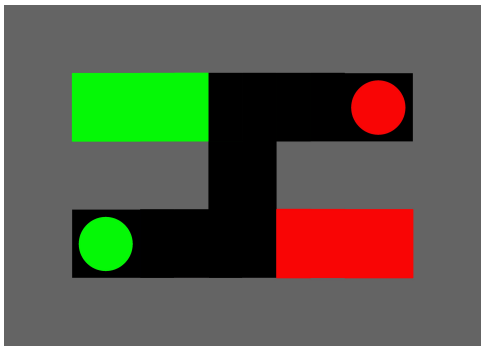
Um die Leistungsfähigkeit von ClusterComm besser einschätzen zu können, dienen die folgenden Ansätze zum Vergleich:

i) Random: Jede Aktion wird mit gleicher Wahrscheinlichkeit zufällig ausgewählt.

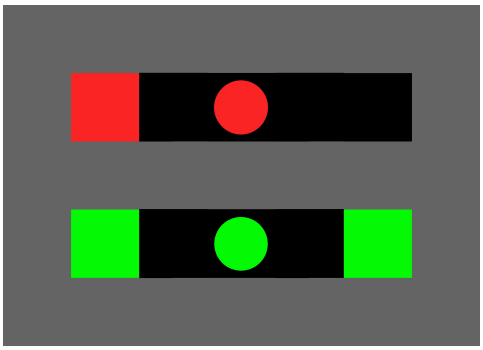
ii) NoComm: Die Agenten werden ohne Kommunikationskanal trainiert.

iii) LatentComm [Lin+21]: Die gesamte Repräsentation wird übermittelt. Dieser Ansatz kann als obere Grenze betrachtet werden, da die Agenten hochdimensionale und kontinuierliche Vektoren übermitteln.

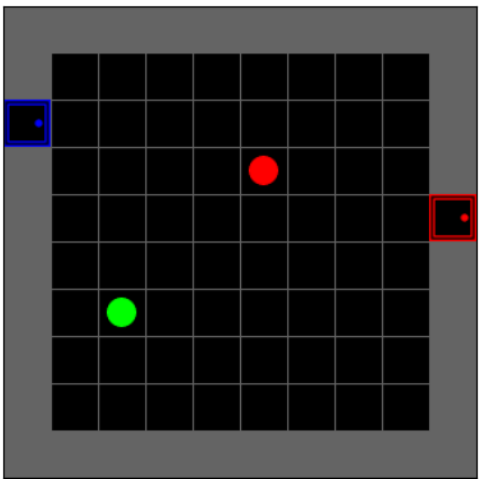
a) Bottleneck



b) ClosedRooms



c) RedBlueDoors



d) Level-based Foraging

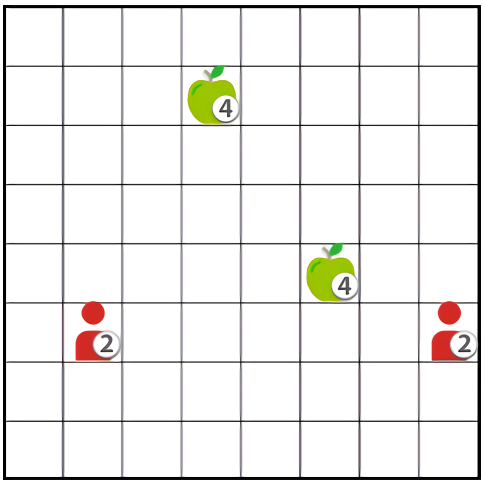


Abbildung 5.2: Visuelle Darstellung der Evaluationsumgebungen.

Das Ziel ist es zu zeigen, dass ClusterComm eine ähnliche Leistung wie LatentComm erreicht, auch wenn die Agenten nur diskrete Nachrichten übermitteln dürfen. Obwohl durch die Anwendung von Clustering möglicherweise nützliche Informationen verloren gehen, ist ClusterComm durch die Reduktion der benötigten Bandbreite besser skalierbar.
Für ClusterComm, seine Variationen sowie die genannten Baselines wird für $\phi_m^{(i)}$ und $\phi_o^{(i)}$ jeweils ein FFNN mit dem Tangens Hyperbolicus Aktivierungsfunktionen und zwei versteckten Schichten der Größe 32 verwendet. Beide Ausgaben werden konkateniert und ein einschichtiges FFNN verwendet, um die Aktionsverteilung vorherzusagen. Darüber hinaus wird K-Means++ nach jedem PPO Update angewandt. Als Eingabe für den Message Encoder dienen immer die letzten drei lokalen Beobachtungen (*Frame Stacking*). Die Anzahl der Cluster liegt für RedBlueDoors und Level-based Foraging bei 16 bzw. 8 bei Bottleneck und ClosedRooms. Abbildung 5.3 dient der Darstellung des Lernverhaltens der einzelnen Ansätze.
Um die Leistung der trainierten Agenten beurteilen zu können, wurden die trainierten Policies 1000 mal evaluiert und der durchschnittlicher Reward, die Erfolgsquote sowie durchschnittliche Anzahl von Schritten berechnet, vgl. Tabellen 5.1 und 5.2.

5.1.3.3 Diskussion der Ergebnisse nach Evaluationsumgebung

In diesem Abschnitt sollen nun die Ergebnisse aus Abbildung 5.3 diskutiert werden. Zur besseren Übersichtlichkeit werden diese im Folgenden für jede Evaluationsumgebung getrennt besprochen.

Bottleneck: Die Ergebnisse für eine unterschiedliche Anzahl von Agenten sind in den Abbildungen 5.3 a) bis c) sowie in Tabelle 5.1 aufgeführt. Bei diesen Experimenten ist vor allem die Skalierbarkeit von ClusterComm in Bezug auf die Anzahl der Agenten von Interesse. Je mehr Agenten hinzukommen, desto schwieriger wird die Aufgabe.
Es ist zu sehen, dass der Unterschied zwischen den Ansätzen mit zunehmender Komplexität, d.h. durch Hinzufügen weiterer Agenten, immer offensichtlicher wird. Mit zwei Agenten schneiden alle Methoden ähnlich gut ab und konvergieren zur optimalen Lösung, vgl. Abbildung 5.3 a). Es zeigt sich jedoch erneut, dass LatentComm die schnellste Konvergenz aufweist, während NoComm mehr Zeit für die Konvergenz benötigt und eine instabilere Lernkurve besitzt. Die Ergebnisse für drei Agenten sind ähnlich. Der Unterschied besteht darin, dass die Methoden mehr Episoden benötigen, um zu konvergieren und dass es NoComm in diesem Fall nicht mehr schafft, eine gleich gute Policy zu finden. Mit vier Agenten werden die Unterschiede zwischen den Ansätzen nochmals größer, vgl. Abbildungen 5.3 c). LatentComm erzielt immer noch die besten Ergebnisse, gefolgt von CentroidComm, ClusterComm und schließlich NoComm. Zudem ist eine größere Varianz als zuvor zu beobachten. Aufgrund des deutlich erhöhten Koordinationsaufwands müssen die Agenten während

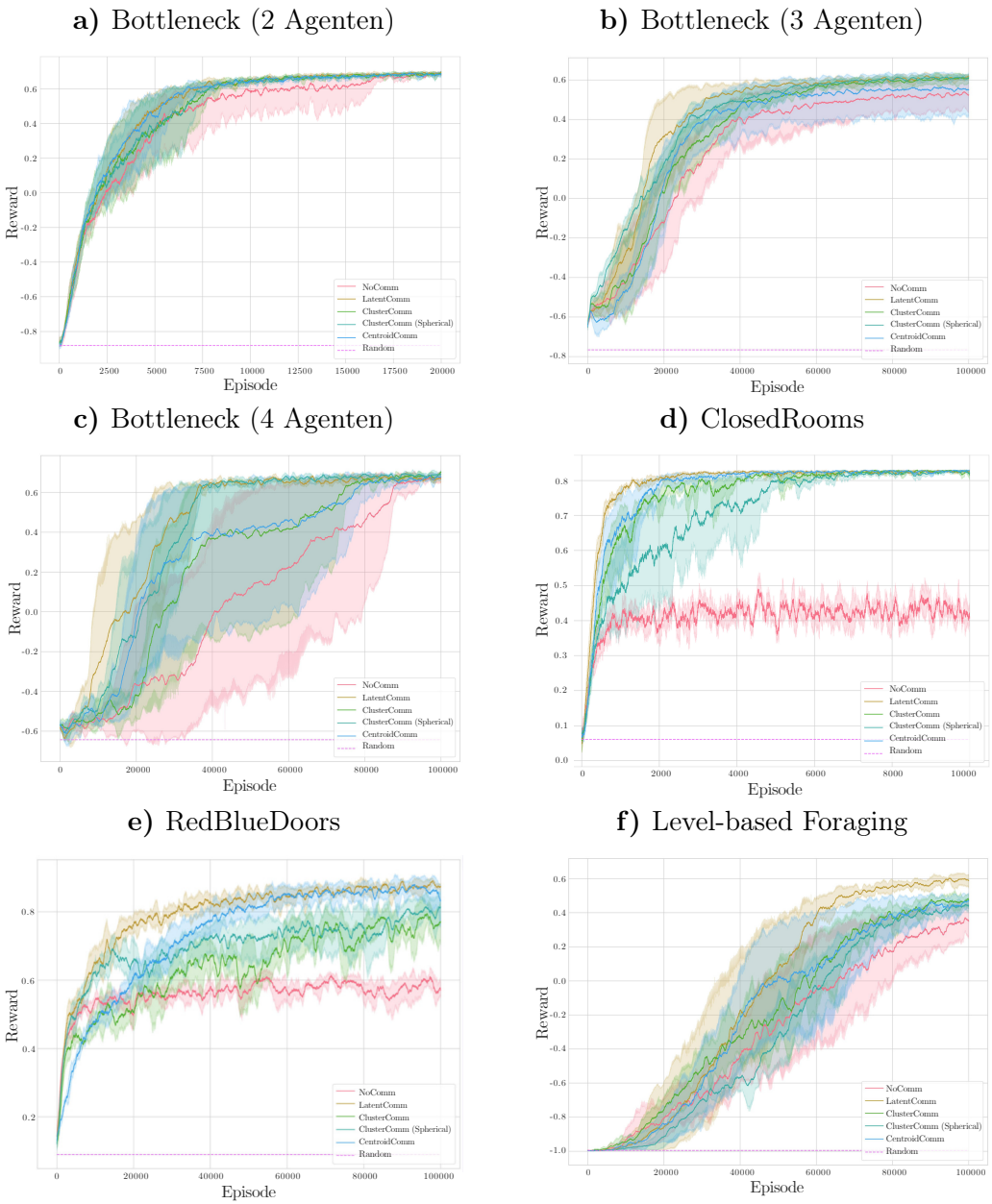


Abbildung 5.3: Lernkurven der Evaluationsumgebungen

Bottleneck									
	2 Agenten			3 Agenten			4 Agenten		
Algorithmus	⌀Rew.	⌀ E.Q.	⌀Schr.	⌀Rew.	⌀ E.Q.	⌀Schr.	⌀Rew.	⌀ E.Q.	⌀Schr.
NoComm	0.70	0.99	9.17	0.52	0.93	12.84	0.67	0.97	18.73
LatentComm	0.70	0.99	9.22	0.60	0.98	12.07	0.67	0.97	19.04
ClusterComm	0.70	0.99	9.39	0.60	0.98	12.21	0.70	0.98	17.78
ClusterComm(S)	0.71	0.99	9.14	0.60	0.98	12.08	0.69	0.97	17.83
CentroidComm	0.71	0.99	9.11	0.55	0.92	11.93	0.67	0.96	18.45
Random	-0.87	0.008	26.11	-0.76	0.0	26.83	-0.64	0.0	55.37

Tabelle 5.1: Durchschnittlicher Reward, Erfolgsquote und durchschnittliche Anzahl von Schritten der trainierten Agenten in der Evaluationsumgebung Bottleneck für eine unterschiedliche Anzahl von Agenten.

des Trainings deutlich mehr explorieren, um schlussendlich eine geeignete Policy zu finden. Auch hier helfen alle ClusterComm Ansätze diese früher und verlässlicher (bzgl. der Varianz) zu finden.

ClosedRooms: Die Ergebnisse sind in Abbildung 5.3 d) dargestellt. Da diese Umgebung Kommunikation erfordert, können Methoden wie Random oder NoComm die Aufgabe nicht lösen. NoComm konvergiert zu der optimalen Strategie, die ohne Kommunikation möglich ist, d.h., der Zuhörer wählt in jeder Episode ein zufälliges Ziel. So können die Agenten das Problem in 50% der Fälle erfolgreich lösen. Andererseits konvergieren die Methoden mit Kommunikation zur optimalen Lösung, d.h. sie lernen, die Problemstellung vollständig zu lösen. Sie unterscheiden sich jedoch in Lerngeschwindigkeit und Stabilität. Es wird ersichtlich, dass LatentComm schneller konvergiert und zusätzlich eine geringere Varianz aufweist. ClusterComm und CentroidComm lernen etwas langsamer, konvergieren aber ebenso zum selben Ergebnis. Darüber hinaus ist die Varianz von ClusterComm geringer als die von CentroidComm. Dies kann auf den Nachrichtentyp zurückgeführt werden. Nach der Aktualisierung des Clusters bleibt der Clusterindex häufig gleich, während sich der Zentroid kontinuierlich ändert. Dies bedeutet, dass sich die ClusterComm-Nachrichten bei den gleichen Beobachtungen seltener ändern, während sich die CentroidComm-Nachrichten nach jeder Aktualisierung ändern, bis das Verfahren konvergiert. In Bezug auf Tabelle 5.2 ergeben sich für alle Methoden mit Kommunikation die gleichen optimalen Ergebnisse (99% Erfolgsrate mit durchschnittlich 3 Schritten pro Episode). NoComm löst dabei nur die Hälfte der Episoden erfolgreich, allerdings mit einer etwas geringeren durchschnittlichen Schrittzahl. Mit NoComm trainierte Agenten wählen zufällig eine Ecke und bewegen sich auf diese zu. Die niedrige Schrittzahl von NoComm kommt also deswegen zustande, weil Agenten sich zwar immer schnell in eine zufällige Ecke bewegen und die Episode terminiert, dies aber nur in der Hälfte der Fälle die richti-

	ClosedRooms			Level-based Foraging			RedBlueDoors		
Algorithmus	⌀Rew.	⌀ E.Q.	⌀Schr.	⌀Rew.	⌀ E.Q.	⌀Schr.	⌀Rew.	⌀ E.Q.	⌀Schr.
NoComm	0.40	0.49	2.60	0.39	0.87	60.29	0.58	0.63	48.09
LatentComm	0.82	0.99	3.05	0.60	0.98	47.52	0.86	0.91	34.20
ClusterComm	0.82	0.99	3.05	0.47	0.92	55.53	0.70	0.75	39.92
ClusterComm(S)	0.82	0.99	3.05	0.46	0.93	57.49	0.77	0.83	36.32
CentroidComm	0.82	0.99	3.04	0.42	0.88	57.58	0.84	0.88	26.00
Random	0.06	0.12	9.18	-0.99	0.00	128.00	0.08	0.16	272.44

Tabelle 5.2: Durchschnittlicher Reward, Erfolgsquote und durchschnittliche Anzahl von Schritten der trainierten Agenten in der Evaluationsumgebung ClosedRooms.

ge Lösung darstellt. Im Gegensatz dazu lernen die Agenten beim Verfahren mit Kommunikation beim ersten Zeitschritt stehenzubleiben und auf die vom Sprecher gesendete Nachricht zu warten.

RedBlueDoors: RedBlueDoors ist die einzige ordinale Umgebung, d.h. der Erfolg ist von der Reihenfolge der ausgeführten Aktionen abhängig.
Auch in Abbildung 5.3 f) zeigt sich deutlich, dass die Möglichkeit der Kommunikation von Vorteil ist. Alle Methoden mit Kommunikation erzielen sehr gute Ergebnisse. LatentComm lernt am schnellsten, gefolgt von CentroidComm. Beide Ansätze konvergieren zu einem optimalen Wert. Die Leistung von ClusterComm ist etwas schlechter, aber immer noch besser als die von NoComm.
Die Auswertung aus Tabelle 5.2 bestätigt, dass LatentComm (Erfolgsquote 91%) und CentroidComm (Erfolgsquote 88%) am besten geeignet sind, gefolgt von ClusterComm, dessen Erfolgsquote bei 75% liegt und mehr Schritte benötigt, um eine Episode abzuschließen.
Auffällig ist ebenfalls, dass CentroidComm im Durchschnitt weniger Schritte beansprucht als LatentComm, obwohl LatentComm eine höhere Erfolgsquote aufweist. Dabei ist zu beobachten, dass CentroidComm in einer erfolgreichen Episode deutlich weniger Schritte braucht als LatentComm und die Agenten sich im Vergleich zu LatentComm in mehr Fällen nicht ausreichend koordinieren. Auf der einen Seite führt die konstante Anpassung der Repräsentationen während des Trainings bei LatentComm folglich dazu, dass die Agenten länger für die Problemlösung brauchen, auf der anderen Seite erhöht sich dadurch deren Erfolgsquote. Durch das Versenden der Zentroide wird die Varianz der einzelnen Merkmale eingeschränkt, da Repräsentationen unterschiedlicher Beobachtungen auf ein und denselben Zentroiden abgebildet werden können. Gleichzeitig ist dies der größte Nachteil von CentroidComm, da so Informationen verloren gehen, was die Erfolgsquote verringert.

Level-based Foraging: Die Komplexität der Umgebung lässt sich anhand

der Ergebnisse der Zufallsagenten beurteilen. Im Gegensatz zu den vorherigen Experimenten zeigt sich, dass die Zufallsagenten einen nahezu minimalen Reward (-1) erhalten, was bedeutet, dass sie das Problem nie lösen.
Es wird auch hier erneut deutlich, dass alle kommunikationsbasierten Ansätze NoComm übertreffen, wobei LatentComm durch sein kontinuierliches Nachrichtenformat die besten Ergebnisse liefert, vgl. Abbildung 5.3 e). Überdies weisen alle ClusterComm Ansätze in etwa die gleiche Leistung auf, wobei ein leichter Vorteil für ClusterComm festzustellen ist, vgl. Tabelle 5.2.
Die Agenten müssen nicht nur die Position der Nahrungsquellen finden, sondern sich auch darauf einigen, zu welcher Nahrungsquelle sich beide Agenten bewegen sollen. Das etablierte Kommunikationsprotokoll hilft den Agenten bei der Entscheidungsfindung.

Die Ergebnisse legen nahe, dass keine der vorgestellten ClusterComm Varianten den anderen deutlich überlegen ist. Deren Leistungsfähigkeit hängt stark von der Umgebung, der darin zu lösenden Problemstellung und der daraus entstehenden Trainingsdynamik ab.

5.1.4 Fazit und mögliche Verbesserungen der Kommunikationsstrategie

In dieser Arbeit wurde ClusterComm vorgestellt, ein MARL-Algorithmus, der effiziente Kommunikation zwischen Agenten ermöglicht. ClusterComm diskretisiert die internen Repräsentationen der eingehenden Beobachtung eines jeden Agenten und verwendet die Clusterindizes als Nachricht. Im Gegensatz zu den meisten anderen Ansätzen lernen die ClusterComm-Agenten unabhängig und ohne die Verwendung von zentralisierten Lernmethoden. Die einzige Voraussetzung ist ein Kommunikationskanal, über den die Agenten Nachrichten austauschen können. In vier verschiedenen Evaluationsumgebungen konnte gezeigt werden, dass ClusterComm dem Ansatz ohne Kommunikation generell überlegen ist. Darüber hinaus ist die Leistungsfähigkeit von ClusterComm in Bezug auf LatentComm konkurrenzfähig. Während LatentComm die Übertragung von kontinuierlichen Vektoren erfordert, muss bei ClusterComm jeweils immer nur ein einziger Integer übertragen werden.
In zukünftigen Arbeiten könnte die Architektur der Netze von der Abhängigkeit von der Anzahl an Agenten befreit werden. Ferner sollte untersucht werden, wie ClusterComm erweitert werden kann, um pro Nachricht mehrere Indizes versenden zu können. Eine umfangreichere Nachrichtengröße könnte dabei helfen, dem Informationsverlust entgegenzuwirken.
Alle aktuellen Ansätze zur Kommunikation im MARL versenden pro Zeitschritt eine Nachricht und führen die nächste Aktion aus. Denkbar wäre es, eine längere Kommunikationsphase zu etablieren, in der die Agenten auf die erhaltenen Nachrichten mit neuen Nachrichten reagieren können. So ließen

sich beispielsweise Unklarheiten durch erneutes Nachfragen beseitigen. Zudem würde die Möglichkeit geschaffen werden, Verhandlungen zu führen, um einen gemeinsamen Konsens zu erreichen.

5.2 Anomalieerkennung im Reinforcement Learning

Die Anomalieerkennung beschäftigt sich mit der Identifikation von Datenpunkten, die erheblich von der Normalität abweichen. Sie wird beispielsweise eingesetzt, um medizinische Probleme, Betrug oder Produktionsfehler zu erkennen. Da bei dieser Problemformulierung zu den Datenpunkten keine zusätzlichen Angaben über den Grad der Abweichung gegeben sind, besteht der vorherrschende Ansatz darin, eine auf einem neuronalen Netz (NN) basierende Bewertungsfunktion zu erlernen, welche den Normalfall modelliert. Typische Datenquellen im aktuellen Forschungskontext der Anomalieerkennung sind z.B. Bilder [Ruf+19; Tac+20; RH21], Video [SCS18; PNH20], Audio [Mül+21d] und Text [Ruf+19].
Anders als beim Reinforcement Learning (RL) sind diese Datenquellen statisch und kommen ohne sequentielle Entscheidungsfindung aus. Im Gegensatz dazu versucht ein Agent im RL ein Belohnungssignal (Reward) zu maximieren, indem er durch Versuch und Irrtum mit seiner Umgebung interagiert. Der Agent und die Umgebung, in der dieser sich bewegt, produzieren also erst im Laufe dieses Lernprozesses Daten wie den aktuellen Zustand, die gewählte Aktion und den Reward.
Es ist davon auszugehen, dass trainierte RL-Agenten in realistischen Szenarien nicht ausschließlich in der Umgebung agieren werden, in der sie trainiert wurden. So können unvorhergesehene Situationen auftreten, wie z.B. unbekannte Verkehrsschilder, die plötzliche Verringerung der Bremsleistung eines autonomen Fahrzeugs, auf die Straße laufende Menschen, das Auftauchen eines Geisterfahrers oder ein verändertes Fahrverhalten anderer Verkehrsteilnehmer.
Da nicht alle möglichen Situationen während des Trainings antizipiert und nachgestellt werden können, wird die Diskrepanz zwischen Trainings- und Einsatzumgebung nicht die Ausnahme, sondern die Regel für RL-Systeme in der realen Welt sein. Ein großer Teil der Forschung lässt diesen Aspekt jedoch außer Acht und trainiert und evaluiert in derselben Umgebung.
Zudem kann der Agent in unbekannten Umgebungen durch seine eigenen Handlungen unbekannte Situationen schaffen, die drastische Veränderungen verursachen. Fährt ein Agent beispielsweise über ein Regal, weil sich die Position des Regals verändert hat, können Produkte aus diesem fallen, den Weg versperren und den Agenten beschädigen. Die Vermeidung solcher Szenarien durch frühzeitige Erkennung möglicher Gefahrensituationen durch Systeme zur Annomalieerkennung ist daher in sicherheitskritischen Szenarien unverzichtbar.
Dennoch ist ein eklatanter Mangel an Forschung auf dem Gebiet der Annoma-

lieerkennung im RL festzustellen. Es dominieren einfache und unrealistische Evaluationsszenarien, welche nicht vollständig auf die einzigartigen Herausforderungen eingehen, die sich aus der RL Problemformulierung ergeben. Dies ist darauf zurückzuführen, dass die Problemstellung in den anderen Feldern bereits eindeutiger formuliert und formalisiert wurde. In den folgenden Abschnitten wird versucht, diese Lücke weiter zu schließen.

5.2.1 Kritische Diskussion von bisherigen Forschungsarbeiten

In diesem Abschnitt werden zunächst verwandte Forschungsarbeiten erörtert und deren Defizite diskutiert. Anschließend wird dargelegt, wie andere Forschungsbereiche ähnliche Fragen stellen und sich mit ähnlichen Themen befassen. Ferner wird gezeigt wie dies dazu beitragen könnte, den Bereich der Anomalieerkennung im Reinforcement Learning auszubauen.

5.2.1.1 Unrealistische Problemstellungen

Für eine fundierte Diskussion über die Herausforderungen, Grenzen und offenen Fragen, die Annomalieerkennung im RL mit sich bringt, ist es wichtig, den aktuellen Stand des Feldes zu überprüfen. Während andere Arbeiten die Bedeutung der Annomalieerkennung als einen der Bausteine für sichere RL-Systeme konzeptionell hervorgehoben haben, gibt es nur sehr wenige Arbeiten, die sich explizit mit der Erkennung von Anomalien in RL in Form von neuen Algorithmen, Domänen oder Evaluations-Szenarien beschäftigen. Darüber hinaus ist festzustellen, dass die meisten bestehenden Arbeiten sich mit eher traditionellen Problemen beschäftigen. Trotz des Anspruchs, RL-spezifische Probleme der Annomalieerkennung zu lösen, beschränken sich die vorgeschlagenen Methoden auf klassische maschinelle Lernaufgaben, bei denen RL nur als Datenquelle dient. Überdies werden die Ansätze in der Regel anhand von unrealistischen und einfachen Szenarien evaluiert. In [Zha+21a] zielen die Autoren beispielsweise darauf ab, fehlerhafte Beobachtungen in einer von einem RL-Agenten gesammelten Trajektorie zu erkennen. Fehlerhafte Beobachtungen wurden durch additives Gaußsches Rauschen, einfache White-Box-Perturbationen oder durch das Ersetzen eines zufälligen Anteils der Beobachtungen in einer Trajektorie durch Beobachtungen aus einer anderen Umgebung erzeugt. Der Anomaliedetektor wird anhand der Merkmalsvektoren der Beobachtung (Aktivierungen der vorletzten Schicht des NN des Agenten) auf der Grundlage der robusten Mahalanobis-Distanz trainiert und berücksichtigt nur einen einzigen Zeitschritt. Dies ist im Grunde die Anwendung eines traditionellen Anomaliedetektors auf einen Datensatz von Merkmalsvektoren (Repräsentationen). Zudem ist es unrealistisch anzunehmen, dass ein Angreifer die Beobachtungen direkt verändern und Gradienten durch den Agenten leiten könnte, um starkes Störrauschen zu erzeugen.

In [Sed+20b] verwenden die Autoren die Entropie (Unsicherheit) der prognostizierten Aktionen zur Anomaliebewertung, um überraschende und ungewöhnliche Zustände zu erkennen. Dies ist dem einfachsten Ansatz zur Out-of-Distribution-Detection bei Klassifizierungsproblemen sehr ähnlich. Bei diesem wird die Entropie über die vorhergesagten Klassen verwendet. Die Methode wird auf einem fixen Satz von prozedural generierten Umgebungen trainiert. Anschließend wird ein weiterer Satz prozedural generierter Testumgebungen generiert und überprüft, ob zwischen den Trainings- und Testumgebungen unterschieden werden kann.
In [Sed+20a] spiegeln die Autoren die Beobachtungen zum Testzeitpunkt vertikal. Die Methoden berücksichtigen jedoch nicht die sequentielle Natur der Beobachtungen. Noch wichtiger ist, dass die Evaluierungsszenarien künstlich und simpel gehalten sind und keine realistischen Bedingungen widerspiegeln. Störrauschen kann sehr leicht simuliert werden, und falls gewünscht, existieren zuverlässige Methoden für das Training von störungsresistenten NNs [For+20]. Die Erkennung von Beobachtungen aus unterschiedlichen Datenquellen führt in den oben genannten Fällen zu visuell sehr unterschiedlichen Beobachtungen (z. B. Objekte, Farben, Textur), die sehr wahrscheinlich mit vortrainierten Netzen erkannt werden könnten. Diese Probleme sind denen der Anomalieerkennung in Videos oder Bildern sehr ähnlich. Abbildung 5.4 veranschaulicht einige der genannten Szenarien.
Der entscheidende Punkt ist, dass diese (einfachen) Probleme leicht auf klassische Mustererkennung reduziert werden können und dass die Arten von Anomalien, die gegenwärtig untersucht werden, keine natürlich vorkommenden Daten und Situationen darstellen.
Folglich müssen anspruchsvollere Herausforderungen entwickelt werden, die den RL-Kontext berücksichtigen und besser auf reale Szenarien ausgerichtet sind.

5.2.1.2 Verwandte Problemstellungen und deren Einschränkungen

Nachdem die Bedeutung der Entwicklung geeigneter Szenarien für die Annomalieerkennung im RL hervorgehoben wurde, soll die Diskussion nun um weitere vielversprechende Szenarien und Ansätze erweitert werden. Dabei werden solche Arbeiten berücksichtigt, die sich nicht explizit mit der Annomalieerkennung befassen, aber potenziell von Interesse für diesen Anwendungsbereich sein könnten.
Eine Reihe von Arbeiten beschäftigt sich mit der Out-of-distribution Generalisierung, wenn die Umwelt-Parameter verändert werden. In diesem Fall sollte der Agent in der Lage sein, eine oder mehrere Aufgaben in neuartigen, aber miteinander verwandten Umgebungen zu erfüllen. Dies ist deshalb wichtig, weil RL-Algorithmen bekanntermaßen unter einer Überanpassung an die Trainingsumgebung leiden [Zha+18a; BPP20]. Zu den betrachteten Szenarien gehören: Eine simulierte Roboterhand, die deutlich kleinere Würfel [Man+19] oder zusätzliche Objekte [EL22] manipulieren muss, welche sie zuvor nicht betrachtet

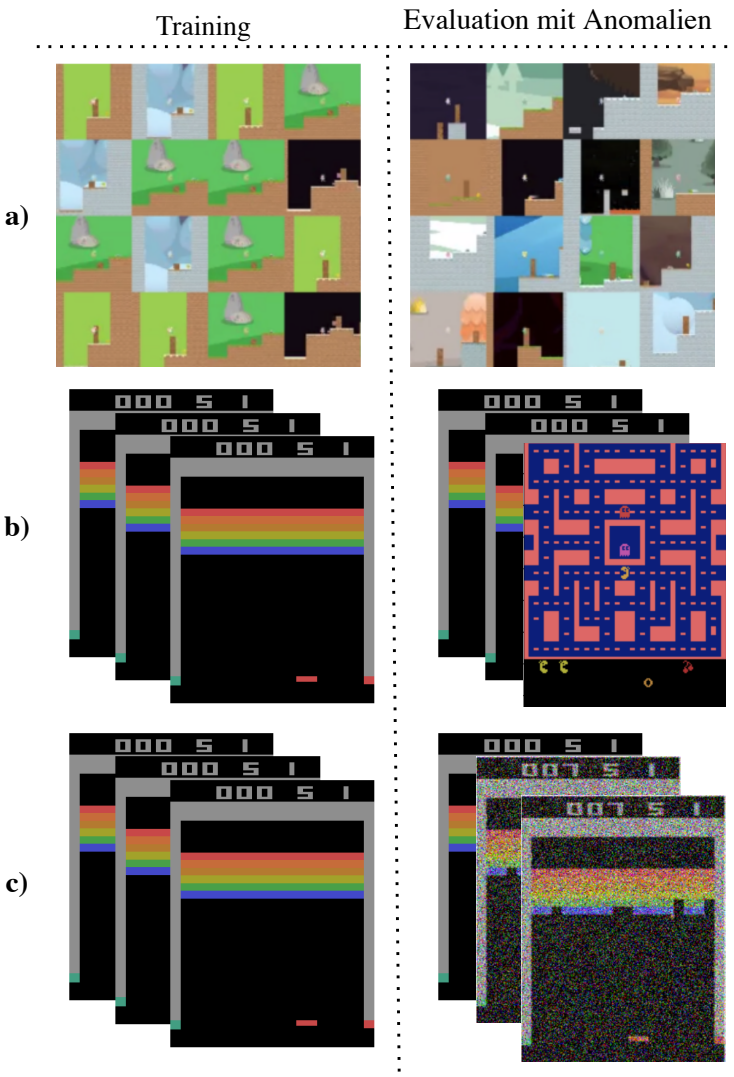


Abbildung 5.4: Veranschaulichung von aktuellen Evaluierungsszenarien der Anomalieerkennung im Kontext des Reinforcement Learnings. a) Der Agent wird auf einem fixen Satz von prozedural generierten Umgebungen trainiert. Anschließend wird ein weiterer Satz prozedural generierter Testumgebungen generiert und überprüft, ob zwischen den Trainings- und Testumgebungen unterschieden werden kann [Sed+20b]. b) Hier werden die Beobachtungen, welche von der Umgebung zu jedem Zeitschritt ausgegeben werden, zum Zeitpunkt der Evaluation zufällig mit Beobachtungen aus anderen Videospielen ersetzt [Zha+21a]. c) Hier wird den Beobachtungen zufällig Gaußsches Rauschen hinzugefügt [Zha+21a].

hat, das Variieren der Länge und Breite der Gliedmaßen eines Agenten zum Testzeitpunkt [Man+21] oder das Modifizieren von Beobachtungen, sodass alternative Hintergrundbilder [Han+20; Wan+20] verwendet werden.
Maximum Entropy RL [EL22] und Self-Supervised Representation Learning [Han+20] haben sich als besonders geeignet erwiesen, die Generalisierung zu verbessern. Ersteres fördert die Exploration und hilft, eine vorzeitige Konvergenz zu suboptimalen Policies zu verhindern, während Letzteres ein Zusatzziel optimiert, das nicht von externen Labels abhängt, um einen robusten Merkmalsraum zu konstruieren, der die Generalisierung unterstützt. Während diese Arbeiten oft nur durch die Veränderung des visuellen Erscheinungsbildes der Beobachtungen evaluiert werden, d.h. durch die Veränderung des Zustandsraumes, was dann im Wesentlichen darauf hinausläuft, Domänenadaptionsansätze aus der Computervision einzusetzen, verändern einige dieser Arbeiten auch die Morphologie der Agenten während der Evaluation [Lee+20; Man+21] oder zielen direkt darauf ab, die Kluft zwischen Simulation und Realität zu überbrücken. Dies bedeutet eine Veränderung der Dynamik sowie des Zustandsraums [Hai+21].
Ob Methoden, die die Generalisierung verbessern, noch erkennen können, ob der Agent mit neuartigen und zuvor unbekannten Situationen konfrontiert ist, oder ob diese Methoden die Erkennungsleistung verschlechtern, ist eine interessante Frage, die es noch zu beantworten gilt.
Danesh et al. [DF21] sind die ersten, die einen Schritt in diese Richtung gehen und das Problem Out-of-Distribution Dynamics Detection (OODD) betiteln. Hier wird die angepasste Umweltdynamik als Anomalie interpretiert, die innerhalb eines kurzen Zeitrahmens erkannt werden muss. Es werden vier verschiedene Arten von Anomalien vorgeschlagen, die auftreten können: Gaußsches Rauschen, Sensorabschaltung (einige Merkmale werden auf null gesetzt), Sensorkalibrierungsfehler (Merkmale werden mit einer Konstanten multipliziert) und Sensordrift (Zunahme des Rauschens im Laufe der Zeit). Es sei angemerkt, dass diese Anomalien ausschließlich auf Rauschen in Umgebungen mit kontinuierlichen, niedrigdimensionalen Beobachtungen basieren. Um die Netze gegenüber den ersten drei Arten von Anomalien widerstandsfähiger zu machen, reichen robuste RL-Ansätze [Pin+17; EL22] oder einfache Datentransformationen aus. Diese Arten der Anomalien sind leicht zu antizipieren. Andererseits ist die Sensordrift eine nicht-stationäre Anomaliequelle. Da die Welt (aus der Sicht des Agenten) inhärent nicht stationär ist, stellt die Nichtstationarität wohl eine der wichtigsten Herausforderungen dar, mit denen zukünftige Annomalieerkennungs-Systeme im Kontext des RLs umzugehen in der Lage sein sollten.
Damit im Zusammenhang steht das Problem des lebenslangen (kontinuierlichen) RL [Khe+20]. Hier müssen Agenten in einer sich ständig ändernden Umgebung und mit ständig wechselnden Aufgaben lernen und sich entsprechend anpassen. Es gibt daher keine klare Trennung zwischen Training und Evaluation. Das Problem selbst ist jedoch durch die Nichtstationarität mo-

tiviert, wobei zwischen aktiver und passiver Nichtstationarität unterschieden wird. Kann der Agent die Art der Nichtstationarität durch sein eigenes Verhalten beeinflussen (z. B. durch das Halluzinieren neuer Ziele), wird von aktiver Nichtstationarität gesprochen, ansonsten von passiver Nichtstationarität.
Um hier Fortschritte zu erzielen und neue Herausforderungen zu schaffen, müssen Szenarien entwickelt werden, in denen Anomalien subtiler und semantisch bedeutsamer sind als einfache, leicht spezifizierte Erweiterungen des Zustandsraums und/oder der Übergangsfunktion. Anomalien müssen so komplex sein, dass sie nicht vorhergesagt werden können. Letztlich könnte die Anomaliequelle selbst von einem anderen RL-Agenten kontrolliert werden.
Dies wird unweigerlich zu weniger allgemeinen Evaluierungsszenarien führen, da komplexe Anomalien von der Domäne abhängig sind.

5.2.2 Generalisierung kontra Anomalieerkennung

Der vorangegangene Abschnitt hat deutlich gemacht, dass die Generalisierung in RL zwar eine ganz andere Zielsetzung verfolgt als die Anomalieerkennung, beide aber im Kern einen ähnlichen Aufbau besitzen.
Ein typischer Ansatz zur Verbesserung der Generalisierung besteht darin, den Agenten invariant gegenüber irrelevanten Eigenschaften der Umgebung zu machen. Dabei wird davon ausgegangen, dass diese eine Störquelle für den RL-Algorithmus darstellen, z.B. ein autonomes Auto auf der Autobahn, das sich nicht um die sich ändernde Umgebung (z.B. Landschaft, Stadt, Wetter) kümmert, während es sich seinem Ziel nähert.
Bei der Domänen-Randomisierung wird schlichtweg mit einer ausreichenden Anzahl von Variationen der Umgebung [Cob+20] trainiert. Dieser Ansatz leidet unter der hohen Komplexität der Stichproben und ist auf einen Simulator angewiesen, der eine Vielzahl von verschiedenen Umgebungen erzeugen kann. Ähnliche Ansätze verwenden Datentransformationen, um künstlich neue Umgebungen zu schaffen.
Andere [Gel+19; Zha+20b] nutzen die Bisimulationsmetrik [GDG03], bei der zwei Zustände bisimilar sind, wenn sie ein ähnliches langfristiges Verhalten aufweisen, d.h. einen ähnlichen erwarteten Reward und ähnliche Dynamik aufweisen.
Das Konzept, alles zu ignorieren, was nicht mit der eigentlichen Aufgabe zusammenhängt, ist in kontrollierbaren Simulationen durchaus sinnvoll. In sicherheitskritischen Szenarien wie dem autonomen Fahren oder intelligenten Fabriken, kann dies aber schädlich sein [Pha+21]. Das Gleiche gilt für die Anpassung an neue Situationen, z.B. durch die Optimierung von Hilfszielen (z. B. Rotationsvorhersage) zur Testzeit mittels Gradientenabstieg [Han+20].
Sicherheitskritische Szenarien umfassen typischerweise mehrere Aufgaben und Sicherheitsvorgaben. *Das Entwerfen eines geeigneten Lernziels in solch komplizierten Szenarien ist meist schwierig oder sogar unmöglich. Spezifikationsfehler können zu pathologischen Situationen führen, in denen der Agent seine Fähig-*

keiten außerhalb der Verteilung der bekannten Umgebungen beibehält, aber das falsche Ziel verfolgt [Lan+22]. Daher stellt das Ignorieren von Informationen, die für das Ziel irrelevant sind, eine sicherheitskritische Gefahr dar. Folglich sind Komponenten zur Annomalieerkennung in RL-Systemen in sicherheitskritischen Bereichen von besonderer Bedeutung.
Ein autonomes Auto, dessen einziges Ziel es ist, auf der Autobahn von a nach b zu fahren, würde so möglicherweise Anzeichen von entstehenden Waldbränden in der umgebenden Landschaft ignorieren. Durch die Annomalieerkennung hingegen könnte die Bedrohung erkannt und an den Fahrer weitergeleitet werden, damit dieser weitere Maßnahmen ergreifen kann.
Die Suche nach einer geeigneten Reaktionsstrategie, um auf erkannte Anomalien zu reagieren, ist ein weiterer wichtiger Aspekt, den es für zukünftige Ansätze zu berücksichtigen gilt, z. B. durch Umschalten auf eine konservativere oder spezialisierte Policy oder durch Übergabe der Kontrolle an einen Menschen. In einem Szenario des lebenslangen Lernens könnte der Agent auf der Grundlage einer Anomaliebewertung entscheiden, ob er sich weiter anpasst oder nicht. Darüber hinaus sollte die Anzahl der falsch-positiven Meldungen minimiert werden, um die Ermüdung durch Alarme (Alert Fatigue) und den häufigen Wechsel der Policies zu reduzieren. Es ist jedoch zu beachten, dass ein schmaler Grat besteht zwischen der Robustheit oder Invarianz des Agenten gegenüber kleinen Veränderungen in der Umgebung und der Notwendigkeit, Anomalien zu erkennen. *Was ist mit „kleinen Änderungen “ gemeint? Was sollte erkannt werden, was kann ignoriert werden?* Am sinnvollsten ist eine Kombination des Besten aus beiden Welten. Der Agent sollte über ausreichende Verallgemeinerungsfähigkeiten verfügen, sodass er unter dem Einfluss neuartiger Dynamiken und Beobachtungen angemessen handeln kann, aber dennoch in der Lage ist, diese Situationen als ungewöhnlich zu erkennen und zu melden.
Domänenwissen kann verwendet werden, um irrelevante Variationsfaktoren („kleine Änderungen “) zu spezifizieren. Letzteres wird von Praktikern durch die Verwendung von sogenannten Operational Design Domain Specifications adressiert, um explizit zu definieren, inwieweit der Agent sicher agieren muss, beispielsweise beim autonomen Fahren unabhängig von Wetter oder Verkehr. Für die Zukunft sind weniger invasive Ansätze erstrebenswert.
Ein häufiges Missverständnis ist die Annahme, dass alle Anomalien gefährlich seien. Diese Entscheidung hängt davon ab, was der Agent unter Sicherheit versteht. Ohne die Möglichkeit, Anomalien zu erkennen, kann diese Entscheidung jedoch nicht getroffen werden.

5.2.3 Formalisierung der Anomalieerkennung im RL

In der bisherigen Diskussion fehlt eine formale Beschreibung der Anomalieerkennung im RL. In diesem Abschnitt wird versucht, das Problem mit einer Variante eines MDP zu verbinden, die den Aspekt der langsam fortschreitenden Dynamik einbezieht - dem Block Contextual Markov Decisi-

on Process (BC-MDP) [Sod+22]. Im Vergleich zu anderen Formalismen wie dem Hidden-Parameter-Block-MDP [Zha+20b] oder dem Dynamic-Parameter-MDP [XHF21] ist er nicht auf eine episodische Umgebung angewiesen. Insbesondere ist es möglich, dass sich die Dynamik innerhalb von Episoden und nicht nur episodenübergreifend ändert. In sicherheitskritischen Szenarien ist es erforderlich, dass Änderungen innerhalb von Zeitschritten und nicht innerhalb von Episoden (wenn die Umwelt überhaupt episodisch ist) erkannt werden können.
Die folgende Definition eines BC-MDP stammt aus [Sod+22].

Definition 5.1: Block Contextual Markov Decision Process

Ein BC-MDP ist durch ein Tupel $(\mathcal{C}, \mathcal{S}, \mathcal{A}, \mathcal{M})$ definiert, wobei $\mathcal{C}$ der Kontextraum, $\mathcal{S}$ der Zustandsraum, $\mathcal{A}$ der Aktionsraum und $\mathcal{M}$ eine Funktion ist, die einen Kontext $c \in \mathcal{C}$ auf MDP-Parameter und den Beobachtungsraum $\mathcal{M}(c) = \{T^c, S^c\}$ abbildet.

Kurz gesagt, der Zustandsraum und die Übergangsdynamik werden durch den Kontext c bestimmt. Die Anomalieerkennung im RL kann dann wie folgt definiert werden:

Definition 5.2: Anomalieerkennung im Reinforcement Learning

Sei $p(c)$ die Dichte von einem $c \in \mathcal{C}$ und $C_{normal}, C_{anomalous} \subset \mathcal{C}$. Ferner ist anzunehmen, dass

$$C_{normal} \cap C_{anomalous} = \varnothing \quad \wedge \quad C_{anomalous} = \mathcal{C} \setminus C_{normal} \tag{5.6}$$

Dann ist das Ziel, ein geeignetes p zu finden, so dass

$$p(c) > p(c') \quad \forall\, (c, c') \in C_{normal} \times C_{anomalous} \tag{5.7}$$

In der Praxis sind für gewöhnlich nur einige $C'_{normal} \subseteq C_{normal}$ gegeben, bei denen der Kontext nicht direkt beobachtbar ist und inferiert werden muss (unter der Annahme der Identifizierbarkeit). Das heißt, der Kontextraum ist nicht bekannt und es besteht Zugriff auf $\mathcal{M}(c)$, nicht aber auf c. Demzufolge wird das Problem so formuliert, dass erkannt werden soll, ob der Agent in einem bekannten oder unbekannten Kontext handelt. Der Einfachheit halber wird der Fall, in dem C'_{normal} durch einen kleinen Anteil anomaler Kontexte kontaminiert ist, nicht berücksichtigt, aber die obigen Ausführungen können einfach um diesen Fall erweitert werden.

5.2.4 Praktische Anforderungen für zukünftige Probleme

Im Folgenden werden verschiedene Desiderata für zukünftige Probleme definiert, die sowohl realistisch als auch so praxisnah sind, dass sie mit gängigen Methoden umgesetzt und untersucht werden können.

I) Ausreichende Verfügbarkeit von Daten zur Modellierung der Normalität
Der Verfügbarkeit von genügend Daten zur Modellierung der Normalität ist von entscheidender Bedeutung und eine allgemeine Grundannahme in vielen Bereichen der Annomalieerkennung. Es wird dabei angenommen, dass normale Daten günstig und leicht zu beschaffen sind, während die Gewinnung anomaler Daten schwierig, teuer, umständlich und in den meisten Fällen schlicht unmöglich ist. Folglich kann nicht erwartet werden, dass ein vollständiger Satz anormaler Daten, der alle denkbaren Ausprägungen von Anomalien enthält, zur Verfügung steht. Dies ist der eigentliche Grund, aus welchem nicht auf das Supervised Learning zugegriffen werden kann.
Im Zusammenhang mit RL bedeutet das etwa, dass Zugriff auf einige wenige Trainingsumgebungen besteht, auf denen der Normalfall sichergestellt werden kann, z. B. dass keine Menschen während des Trainings die Straße überqueren.

II) Anomalien sind semantisch in der Umgebung verwurzelt
Dieser Punkt wurde bereits in den vorangegangenen Abschnitten eingehend erörtert.
Es ist notwendig, von einfachen Mustererkennungsproblemen abzurücken und semantisch bedeutsame Anomaliequellen zu untersuchen, die tief in der untersuchten Umgebung verwurzelt sind und Nichtstationarität aufweisen. Zum Beispiel die sporadische Präsenz von Straßenarbeitern oder unbekannten Verkehrsschildern zum Zeitpunkt der Evaluation. Ein ähnlicher Trend ist in der Computervision zu beobachten [Dee+21]. Darüber hinaus könnten auch Unregelmäßigkeiten in die Nichtstationarität selbst eingeführt werden, etwa durch Variation des Grades der Nichtstationarität.

III) Der Reward ist nach dem Training nicht mehr verfügbar
Dieser Punkt ist eng verwandt mit dem Problem der skalierbaren Überwachung [Amo+16]. In realen Szenarien ist es oft nicht möglich, die Handlungen des Agenten nach dem Training weiterhin zu belohnen. Dies ist z.B. der Fall, wenn menschliche Aufsicht erforderlich ist oder wertvolle Ressourcen wie externe Überwachungssysteme involviert sind. Aus diesem Grund ist es wesentlich realistischer, wenn der Reward nach dem Training nicht mehr zur Verfügung steht. Ist die Belohnung zur Testzeit verfügbar, liefert ein statistischer Vergleich der Belohnungen aus der Trainings- und Testzeit gute Ergebnisse [Man+21]. Diese Konstellation wird hier aufgrund der fehlenden Relevanz ausgeschlossen.

IV) Simultanes oder post-hoc, Whitebox oder Blackbox Training
Hinsichtlich des Trainings dedizierter Module der Annomalieerkennung kommen aufgrund der RL Problemformulierung mehrere Möglichkeiten infrage:

i) Training des Anomalieerkennungsmodells zusammen mit dem Agenten: Die Herausforderung besteht darin, die Lerndynamik des Agenten zu berücksichtigen. Es ist zu beachten, dass es möglich ist, die inhärenten Eigenschaften wie die Ungewissheit über Handlungen zu nutzen.

ii) Post-hoc-Training des Detektors: Hier sind zusätzliche Durchläufe mit den trainierten Agenten in der Trainingsumgebung erforderlich. Die Herausforderung besteht darin, zu bestimmen, wie viele Durchläufe erforderlich sind, um genügend Daten für die Modellierung der Normalität zu erhalten, und wie dies effizient durchgeführt werden kann. Die Post-hoc-Methode schafft eine Quelle der Redundanz und erhöht die tatsächliche Trainingsdauer, was problematisch sein kann, wenn die Durchläufe teuer sind. Sie könnte jedoch das Problem der Berücksichtigung der Lerndynamik des Agenten reduzieren.

iii) Blackbox-Training: In dieser Konstellation können nur die vom Agenten emittierten und durch die Umgebung erhaltenen Daten beobachtet werden, d.h. Zustände, Aktionen und nächste Zustände. Insbesondere besteht kein Zugriff auf interne Zustände wie die Aktivierungen des NN des Agenten.
Ein Roboterhersteller könnte sich beispielsweise dafür entscheiden, keinen Zugriff auf das NN zu gewähren, da dies sein entscheidendes Verkaufsargument preisgeben würde.

iv) Whitebox-Training: Im Gegensatz zu (iii) sind hier alle Teile des Agenten (z. B. seine Aktivierungen) zugänglich und es ist möglich, zusätzliche Trainingsphasen und -ziele hinzuzufügen, z. B. um zusätzliche Optimierungsziele zur Verbesserung der Erkennungsleistung zu definieren.

V) Minimierung von Falsch-Positiven/Negativen und der Erkennungszeit
Das Ziel sollte eine niedrige Falsch-Positiv-Rate sein, um einen geregelten und praktikablen Betrieb zu gewährleisten und der Alarmmüdigkeit entgegenzuwirken. Die Minimierung falsch negativer Ergebnisse stellt sicher, dass keine Anomalien übersehen werden. Das richtige Gleichgewicht zu finden, ist eine der größten Herausforderungen. Wichtig ist darüber hinaus, dass anomale Situationen ausgehend von möglichst wenigen Interaktionen erkannt werden, d. h. die Erkennungszeit sollte möglichst gering gehalten werden.

VI) Entwurf einer geeigneten Reaktionsstrategie
Unter der Annahme, dass ein zuverlässiger Anomaliedetektor zur Verfügung steht, stellt sich als Nächstes die Frage, wie Situationen zu handhaben sind,

die als anomal gekennzeichnet wurden (Ideen dazu in Abschnitt 5.2.2). Diese Frage ist mutmaßlich sehr domänenspezifisch und birgt ihre eigenen Herausforderungen. Gleichwohl ist diese Fragestellung beim Einsatz von Agenten in der realen Welt unumgänglich.

5.2.5 Lösungsansätze

Nachfolgend werden einige mögliche Lösungen kurz skizziert, die in Zukunft erforscht werden sollten.
Naheliegend ist die Frage, wie und inwieweit Explorationsansätze zur Aufdeckung von anomalen Zuständen oder Situationen umfunktioniert werden können. So könnten beispielsweise die Entropie zufälliger Zustandsmerkmale [Seo+21], der Destillationsfehler bei der Regression der Merkmale eines anderen zufälligen Netzwerks [Bur+18], die Unsicherheit über intermediäre Aktionen aus einem inversen Neugiermodul [Pat+17] oder dichtebasierte Pseudozählungen [ZT19] verwendet werden. Diese Methoden befassen sich jedoch nur mit den Folgen von kurzfristigen Handlungsabfolgen.
Modellbasiertes RL (MBRL) versucht, explizit ein Modell der Komponenten des zugrundeliegenden MDPs zu konstruieren (Weltmodell). Diese Modelle können verwendet werden, um zu überprüfen, ob die erhaltenen Beobachtungen und Transitionen mit dem gelernten Modell der Welt übereinstimmen, z.B. indem die Vorhersagen mit dem tatsächlichen Ereignis verglichen werden. Zusätzlich könnten mit dem Zugang zu anomalen Situationen neuartige anomale Situationen unter Verwendung des Weltmodells des Agenten erschaffen werden [Bal+21]. Allerdings sind MBRL-Ansätze immer noch sehr instabil und schwer zu optimieren.
Eine weitere aussichtsreiche Richtung ist die direkte Ableitung von *Normalitätskontexten*. In der Computervision hat die Idee, Repräsentationen durch selbstüberwachtes Lernen ohne externe Überwachung zu erlernen und die Annomalieerkennung auf diese Repräsentationen anzuwenden, in letzter Zeit an Bedeutung gewonnen [Ruf+18; RH21]. Der Agent könnte mit ähnlichen Repräsentationslernfähigkeiten ausgestattet werden, um während des Trainings einen sinnvollen Raum von Kontextrepräsentationen abzuleiten. Anschließend kann der Raum normaler Kontexte post-hoc kompakter gemacht werden, sodass anomale Kontexte weiter vom Zentrum des Repräsentationsraums entfernt platziert werden. Die Umfunktionierung von Ansätzen, die sich mit Konzeptdrift und Konzeptverschiebung in Zeitreihen befassen, kann sich als weiterer fruchtbarer Weg erweisen. Hier besteht die Herausforderung darin, die Komponente der Entscheidungsfindung des RL mit einzubeziehen.
Schließlich könnte die Ableitung der kausalen Struktur der Umgebung [Zha+20a; Loc+20] zu weiteren Erkenntnissen über die Ursache der Anomalie führen und dazu beitragen, die Annomalieerkennung interpretierbarer zu machen.

5.2.6 Fazit und weitere Fragestellungen

In dieser Arbeit wurde die Bedeutung der Erkennung von Anomalien im Kontext des RL diskutiert und festgestellt, dass die derzeitigen Ansätze und Evaluierungsszenarien nicht hinreichend realistisch sind. Das Problem wurde mit verschiedenen anderen Bereichen des RL in Verbindung gebracht, wie z.B. lebenslangem Lernen und Generalisierung. Zusätzlich wurden konkrete Empfehlungen zur Gestaltung realistischerer Szenarien gegeben. Weiterhin wurden erste Schritte zur Formalisierung des Problems unternommen und verschiedene Ideen diskutiert, wie zukünftige Arbeiten das Problem angehen könnten.
Anomalieerkennung im MARL ist ein weiteres Forschungsgebiet, das eine weitere Komplexitätsebene hinzufügt. Anomalien können in der Kommunikation zwischen Agenten, in ihrer Fähigkeit zur Kooperation oder durch den Einsatz von gegnerischen Agenten entstehen.

5.3 Repräsentationslernen von Fußballteams

Das Feld der datengetriebenen Sportanalyse leidet unter der unzureichenden Verfügbarkeit von kostenlosen und frei zugänglichen Daten. Während nordamerikanische Sportarten wie Basketball, Football und Baseball bereits seit langem Gegenstand der Datenanalyse sind, hat die Sportanalyse insbesondere im Fußball erst in den letzten Jahren an Bedeutung gewonnen.
Maschinelles Lernen verspricht in diesem Bereich die Entscheidungsfindung zu verbessern und zu beschleunigen, z.B. für die taktische Ausrichtung auf das kommende Spiel oder das Erfinden neuer Spielzüge.
Wie in den vorangegangenen Kapiteln bereits besprochen, stellen Merkmalsvektoren bzw. Repräsentationen einen der Grundbausteine des maschinellen Lernens dar. Im Fall der Fußballanalyse sind diese Repräsentationen jedoch schwer zu beschaffen. Um beispielsweise nicht-triviale Merkmale für einen Fußballspieler oder eine Mannschaft zu sammeln, müssen Daten von einem Sportanalyse-Unternehmen gekauft werden, welches gut ausgebildete Experten für die manuelle Datenerfassung beschäftigt.
Um die Abhängigkeit von teuren Experten zu verringern und exemplarisch darzulegen, wie selbst aus geringen, aber frei verfügbaren Datenquellen hochwertige Repräsentationen gelernt werden können, wird in diesem Abschnitt STEVE - **S**occer **TE**am **VE**ctors vorgestellt.
STEVE ist eine Methode zum automatischen Lernen von Repräsentationen von Fußballmannschaften und beruht ausschließlich auf frei verfügbaren Spielergebnissen aus verschiedenen Fußballligen und Wettbewerben. Auf diese Weise wird das Problem der unzureichenden Verfügbarkeit von Daten entschärft. Die mittels STEVE gelernten Repräsentationen können als Eingabe für verschiedene Probleme des maschinellen Lernens wie der Klassifizierung oder der Regression verwendet werden.
In dem sich ergebenden Vektorraum liegen ähnliche Mannschaften nahe beiein-

ander. Da keine einheitliche Definition der Ähnlichkeit zwischen Fußballmannschaften existiert, basiert STEVE auf insgesamt vier eigenen Hypothesen (Abschnitt 5.3.2). Die wichtigste davon besagt, dass zwei Mannschaften ähnlich sind, wenn sie häufig gegen die gleichen Gegner gewinnen. Daher kann *STEVE* verwendet werden, um ähnliche Mannschaften zu finden, indem die Distanz zwischen deren Repräsentationen berechnet wird, und um eine Rangordnung in einer selbst gewählte Liste von Mannschaften zu berechnen.

5.3.1 Repräsentationslernen im Kontext der Sportanalyse

Das Lernen von reellwertigen Repräsentationen für (diskrete) Einheiten ist von besonderem Interesse auf dem Gebiet der Computerlinguistik. Einheiten sind dabei in der Regel Wörter oder Sätze und ihre Repräsentationen werden so gelernt, dass deren Semantik im resultierenden Vektorraum erhalten bleibt. Moderne Ansätze lernen typischerweise eine verteilte Repräsentation für Wörter [Ben+03], basierend auf der distributionellen Hypothese [S H54]. Diese besagt, dass Wörter mit ähnlichen Bedeutungen oft im selben Kontext vorkommen. Der Kontext sind dabei die anderen Wörter im Umfeld des Wortes.
Mit word2vec stellten Mikolov et al. [Mik+13a; Mik+13b] ein neuronales Sprachmodell vor, das die Skip-Gram-Architektur zum Trainieren von Wortrepräsentationen verwendet. Ausgehend von einem zentralen Wort maximiert word2vec iterativ die Wahrscheinlichkeit der Beobachtung des umgebenden Fensters von Kontextwörtern. Die resultierenden Repräsentationen können zur Messung der semantischen Ähnlichkeit zwischen Wörtern verwendet werden. Nach word2vec ist football das ähnlichste Wort zu soccer. Außerdem können mittels Vektorarithmetik Analogien berechnet werden. Sei $r(\cdot)$ eine Funktion, die für ein Wort die entsprechende Repräsentation zurückgibt, dann lässt sich damit folgende Analogie berechnen: *r(König) - r(Mann) + r(Frau) = r(Königin)*. Das Konzept wurde seither auf Graphen erweitert, um eine Repräsentation für jeden Knoten zu lernen. Perozzi et al. [PAS14] und Dong et al. [DCS17] generieren dazu Sequenzen von Knoten durch Random Walks auf dem Graph und interpretiert diese als Sätze. Dies basiert auf der Annahme, dass diese Random Walks als Stichproben aus einem Sprachgraphen interpretiert werden können. Die resultierenden Sätze werden anschließend mit word2vec weiterverarbeitet. Aufbauend auf dieser Idee erstellt LinNet [Pel18] ein gewichteten und gerichteten Match-Up-Graph, dessen Knoten Aufstellungen von NBA-Basketballteams darstellen. Eine Kante von Knoten i zu Knoten j wird eingefügt, wenn die Aufstellung i bessere Resultate erzielte als die Aufstellung j, wobei das Kantengewicht auf die Differenz der Resultate gesetzt wird. Repräsentationen für Aufstellungen werden berechnet, indem node2vec [GL16] auf den resultierenden Graph angewendet wird. Anschließend wird ein logistisches Regressionsmodell auf der Grundlage der zuvor berechneten Repräsentationen erlernt, um die Wahrscheinlichkeit zu modellieren, dass

die Aufstellung λ_i besser abschneidet als die Aufstellung λ_j.
In letzter Zeit wurden die oben genannten Erkenntnisse auf weitere Bereiche der Sportanalyse angewandt.
(batter|pitcher)2vec [Alc16] berechnet Repräsentationen für Baseball Spieler, indem das Ergebnis eines At-Bats für einenen gegebenen Schlagmann (Batter) und Werfer (Pitcher) vorhersagt. Die resultierenden Repräsentationen werden verwendet, um Werfer nach deren Wurfstil zu gruppieren und um zukünftige At-Bat-Ergebnisse vorherzusagen.
Das datengesteuerte Ghosting-Modell von Le et. al [LPY17] basiert auf den Bewegungsdaten einer Saison einer professionellen Fußballliga und modeliert zunächst die defensiven Bewegungsmuster eines durchschnittlichen Teams. Um die strukturellen und stilistischen Elemente eines Teams zu berücksichtigen, wird zusätzlich zu dem league average-Modell für jedes Team eine Teamidentitätsrepräsentation gelernt.
STEVE zielt darauf ab, Repräsentationen für Fußballmannschaften zu lernen und ist daher eng mit den vorgestellten Ansätzen verwandt. Da ein Anwendugsszenario von STEVE die Bewertung von Fußballmannschaften ist, sollen im folgenden kurz ähnliche Ansätze diskutiert werden.
Neumann et al. [NRB18] schlagen eine Alternative zu klassischen ELO- und Pi-Rating-basierten Team-Ranking-Ansätzen vor [HA10; CF13]. Ein Graph, der auf den Spielergebnissen und einer verallgemeinerten Version der sozialen Agonie [Gup+11] basiert, wird verwendet, um Teams in eine geringe Anzahl von diskrete Stufen der Spielstärke zu kategorisieren. Die allgemeine Match-Up-Modellierung wird durch das Blade-Chest-Modell [CJ16] behandelt. Jeder Spieler wird durch zwei d-dimensionale Vektoren dargestellt, den Blade- und Chest-Vektoren. Team a hat gewonnen, wenn sein Blade Vektor näher an dem Chest Vektor von Team b liegt als umgekehrt. Die Intransitivität wird explizit modelliert, indem sowohl Blade- als auch Chest Vektoren verwendet werden, was bei Ansätzen, die jedem Team einen einzelnen skalaren Wert zuordnen, nicht berücksichtigt werden kann [BT52]. Auf ähnliche Weise nutzt auch STEVE zwei Vektoren pro Mannschaft, eine Siegerrepräsentation und eine Verliererrepräsentation.

5.3.2 STEVE: Soccer Team Vectors

In diesem Abschnitt wird *STEVE - Soccer Team Vectors* genauer vorgestellt. Zunächst wird ein Überblick über die zugrunde liegende Idee gegeben und das Ziel des Ansatzes präzisiert. Anschließend folgen die Diskussion der Problemstellung sowie die Vorstellung des Algorithmus.

5.3.2.1 Zugrundeliegende Annahmen

STEVE zielt darauf ab, sinnvolle und nützliche Repräsentationen für Fußballmannschaften zu lernen. Wenn zwei Mannschaften ähnlich sind, sollten ihre Repräsentationen im Vektorraum nahe beieinander liegen, während ungleiche

Mannschaften eine höhere Distanz aufweisen sollten. Darüber hinaus können die Repräsentationen als Merkmalsvektoren für verschiedene maschinelle Lernaufgaben wie der Klassifikation und Regression verwendet werden. Da keine eindeutige Definition zur Ähnlichkeit von Fußballmannschaften existiert, basiert STEVE auf den folgenden vier Annahmen zu deren Ähnlichkeit:

Definition 5.3: Ähnlichkeit von Fußballmannschaften

(i) Die Ähnlichkeit kann unter Berücksichtigung der Spiele, die sie in der Vergangenheit bestritten haben, bestimmt werden.

(ii) Häufige Unentschieden zwischen zwei Mannschaften deuten darauf hin, dass sie ungefähr gleich stark sind.

(iii) Zwei Mannschaften sind ähnlich, wenn sie häufig gegen die gleichen Gegner gewinnen.

(iv) Spiele aus jüngerer Vergangenheit haben einen größeren Einfluss auf die Ähnlichkeit als solche, die lange zurückliegen.

Da die Datenerfassung insbesondere im Bereich der Sportanalytik teuer und zeitaufwändig ist, wird STEVE's Datenbedarf bewusst so gering wie möglich gehalten. Genauer gesagt, werden nur Spieldaten verwendet, welche Informationen darüber enthalten, welche Mannschaften gegeneinander gespielt haben, in welcher Saison ein Spiel stattgefunden hat und ob die Heimmannschaft gewonnen oder verloren hat, oder das Spiel unentschieden ausgegangen ist. Es ist zu beachten, dass die oben genannten Annahmen zur Ähnlichkeit von Fußballmannschaften keine weiteren Informationen erfordern und daher für diesen Fall gut geeignet sind.

5.3.2.2 Problemdefinition

Zur Vereinfachung der Definitionen sei $M = \{1, 2, \ldots, m\}$. Ferner wird angenommen, dass jeder der m Fußballmannschaften, für die eine Repräsentation gelernt werden soll, ein eindeutiger Index $i \in M$ zugeordnet ist. Sei $\Phi \in \mathbb{R}^{m \times \delta}$, wobei jede Zeile Φ_i die δ-dimensionale Repräsentation der Mannschaft i darstellt. Das Ziel ist es, Φ so zu finden, dass $dist(\Phi_i, \Phi_j)$ für ähnliche Teams i und j klein ist und $dist(\Phi_i, \Phi_k)$ für ungleiche Teams i und k groß ist. $dist(\cdot, \cdot)$ ist eine Distanzmetrik, wobei die Ähnlichkeit zwischen den Teams gemäß der Annahmen in Abschnitt 5.3.2.1 bestimmt wird. Des Weiteren wird angenommen, dass die Daten in der folgenden Form vorliegen:

$$\mathcal{D} = \{(a, b, s, d) \in M \times M \times \{1, \ldots, x_{max}\} \times \{0, 1\}\} \tag{5.8}$$

Das Quadrupel (a, b, s, d) steht für ein einzelnes Spiel zwischen den Mannschaften a und b, $s \in \{1, \dots x_{max}\}$ und sagt aus, in welcher der x_{max} Saisons das Spiel stattfand. $d \in \{0, 1\}$ ist ein Indikator der angibt ob das Spiel unentschieden endete ($d = 1$). Wenn $d = 0$, ist das Quadrupel immer so angeordnet, dass Mannschaft a gegen Mannschaft b gewonnen hat.

5.3.2.3 Algorithmus

Gemäß der ersten Annahme kann Φ angepasst werden, während in einer Schleife über den Datensatz $\mathcal{D}$ iteriert wird. Wenn $d = 1$ (unentschieden), wird den Abstand zwischen Φ_a und Φ_b minimiert, wodurch die zweite Annahme berücksichtigt wird. Die dritte Annahme behandelt Beziehungen höherer Ordnung, bei der Mannschaften, die häufig gegen dieselben Mannschaften gewinnen, ähnlich sind. Eine zweite Matrix $\Psi \in \mathbb{R}^{m \times \delta}$ enthält zeilenweise die Verlierer-Repräsentationen (Ψ_i von Team i). Folglich wird Φ_i von nun an als Gewinnerrepräsentation von Mannschaft i bezeichnet. Die Unterscheidung zwischen Gewinner- und Verlierrepräsentation ist unumgänglich, da Distanzmetriken symmetrisch sind. Beide Matrizen Φ und Ψ werden gemäß einer Normalverteilung mit Mittelwert Null und Einheitsvarianz initialisiert. Wenn Mannschaft a gegen Mannschaft b gewinnt, wird der Abstand zwischen Φ_a und Ψ_b minimiert, wodurch sich die Verliererrepräsentationen von b und die Gewinnerrepräsentation von a einander annähern. Das heißt, dass die Verliererrepräsentationen aller Mannschaften, gegen die a häufig gewinnt, in unmittelbarer Nähe der Gewinnerrepräsentation von Mannschaft a liegen werden. Wenn also andere Mannschaften ebenfalls häufig gegen diese Mannschaften gewinnen, müssen ihre Gewinnerrepräsentationen nahe beieinander liegen, um den Abstand zu den Verliererrepräsentationen zu minimieren. Die Parameter Φ und Ψ werden mit Hilfe des stochastischen Gradientenabstiegs mit fogendem Minimierungsziel geschätzt:

$$\arg\min_{\Phi,\Psi} \sum_{(a,b,s,d)\in\mathcal{D}} d * dist(\Phi_a, \Phi_b) + (1 - d) * dist(\Phi_a, \Pi_b) \tag{5.9}$$

Der Abstand zwischen Φ_a und Φ_b wird direkt minimiert, wenn beide Teams unentschieden spielen ($d = 1$). Andernfalls ($d = 0$) wird den Abstand zwischen Φ_a (Gewinnerrepräsentation) und Ψ_b (Verliererrepräsentation) minimiert. Sei nun $dist(\cdot)$ die quadratische euklidische Distanz, dann kann der Ausdruck wie folgt umgeschrieben werden.

$$\arg\min_{\Phi,\Psi} \sum_{(a,b,s,d)\in\mathcal{D}} d * ||\Phi_a - \Phi_b||^2 + (1 - d) * ||\Phi_a - \Pi_b||^2 \tag{5.10}$$

In der obigen Form werden Spiele, die in weit zurückliegenden Spielzeiten ausgetragen wurden, genauso gewichtet wie erst kürzlich ausgetragene Spiele. Das Problem lässt sich entschärfen, indem Spiele aus älteren Saisons mit Hilfe des linearen Gewichtungsschemas $\frac{s}{max(M)}$ weniger stark gewichtet werden. So er-

gibt sich die finale Formulierung des Minimierungsziels:

$$\arg\min_{\Phi,\Psi} \sum_{(a,b,s,d)\in\mathcal{D}} \frac{s}{max(M)}\Big[d * \|\Phi_a - \Phi_b\|^2 + (1-d) * \|\Phi_a - \Psi_b\|^2\Big] \quad (5.11)$$

Dieser Ansatz hat den Vorteil, dass kein manuelles Feature Engineering betrieben werden muss. Alle Annahmen über die Ähnlichkeit zweier Fußballmannschaften sind in den Repräsentationen enthalten. Algorithmus 2 beschreibt die einzelnen Schritte im Detail. Zusätzlich wird der Ansatz in Abbildung 5.5 dargestellt. Zur Veranschaulichung werden die Gradienten nach der Beobachtung eines einzigen Datenpunktes berechnet und auf den Regularisierungsterm verzichtet. Die konkrete Implementierung berechnet die Gradienten über eine größere Menge (Batches) von Datenpunkten, um eine verlässlichere Schätzung des tatsächlichen Gradienten zu erhalten.
In den Zeilen 9, 12 und 15 werden die Repräsentationen auf die Einheitssphäre projiziert, sodass die Länge der Repräsentationen (Vektoren) immer 1 ergibt. Anstatt den Repräsentationen zu erlauben, sich beliebig auszubreiten, hat das Festlegen der zugrundeliegenden Geometrie den Vorteil, dass einzelne Dimensionen die Distanzberechnung nicht dominieren können.

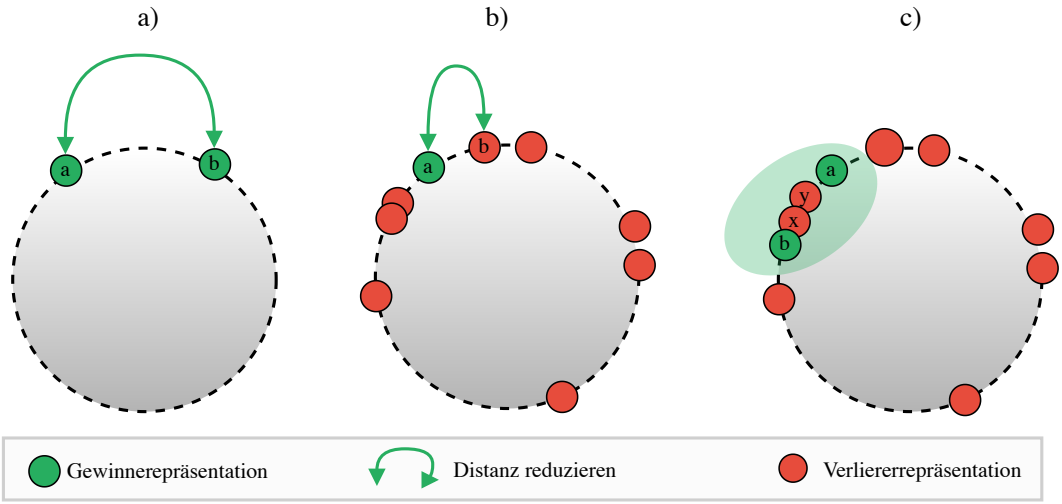


Abbildung 5.5: Veranschaulichung des Ansatzes. a) Die Distanz zwischen den Gewinnerräpräsentationen von Mannschaft a und b wird direkt minimiert da sie unentschieden gespielt haben.b) Mannschaft a hat gegen Mannschaft b gewonnen, somit wird die Distanz zwischen der Gewinnerrepräsentation von a und der Verliererrepräsentation von b reduziert. c) Hier wird veranschaulicht, dass die Distanz zwischen den Gewinnerrepräsentationen von a und b gering ist, wenn sie sich die Nachbarschaft zu ähnlichen Verliererrepräsentationen teilen (bzw. häufig gegen diese gewinnen, hier x und y).

Algoritmus 2 STEVE($\mathcal{D}$, m, δ, α, $max(M)$,e)

$\Phi \sim \mathcal{N}(0,1)^{m \times \delta}$ ▷ Initialisiere Φ
$\Psi \sim \mathcal{N}(0,1)^{m \times \delta}$ ▷ Initialisiere Ψ
for i in $\{1, \ldots, e\}$ **do**
  $\mathcal{D}$ = shuffle($\mathcal{D}$) ▷ Datensatz mischen
  for each (a, b, s, d) in $\mathcal{D}$ **do**
    $L(\Phi, \Psi) = \frac{s}{max(M)}\Big[d * \|\Phi_a - \Phi_b\|^2 + (1-d) * \|\Phi_a - \Psi_b\|^2\Big]$ ▷ Ziel
    if $d = 0$ **then** ▷ a hat Spiel gewonnen
      $\Psi_b = \Psi_b - \alpha * \frac{\partial L}{\partial \Psi_b}$ ▷ Gradientenabstieg auf b's Verl. R.
      $\Psi_b = \Psi_b / \|\Psi_b\|$ ▷ b's Verliererrepräsentation normalisieren
    else ▷ Unentschieden
      $\Phi_b = \Phi_a - \alpha * \frac{\partial L}{\partial \Phi_b}$ ▷ Gradientenabstieg auf b's Gew. R.
      $\Phi_b = \Phi_b / \|\Phi_b\|$ ▷ b's Gewinnerepräsentation normalisieren
    end if
    $\Phi_a = \Phi_a - \alpha * \frac{\partial L}{\partial \Phi_a}$ ▷ Gradientenabstieg auf a's Gew. R.
    $\Phi_a = \Phi_a / \|\Phi_a\|$ ▷ a's Gewinnerepräsentation normalisieren
  end for
end for
return Φ, Ψ

5.3.3 Experimentelle Untersuchung von STEVE

Dieser Abschnitt dient der Schaffung eines Überblicks über den verwendeten Datensatz sowie der Vorstellung und Diskussion verschiedener Experimente, um die Aussagekraft und Wirksamkeit des Ansatzes zu untersuchen.

5.3.3.1 Datensatz und Versuchsaufbau

Der Datensatz besteht aus allen Spielen der Bundesliga (Deutschland), der Premier League (Großbritannien), der Serie A (Italien), der La Liga (Spanien), der Eredivisie (Niederlande), der League 1 (Frankreich), der Süper Lig (Türkei), der Pro League (Belgien), der Liga NOS (Portugal), der Europa League und der Champions League, die von 2010 bis 2019 gespielt wurden. Es wurden insgesamt 29529 Spiele zwischen 378 verschiedenen Mannschaften ausgetragen, von denen etwa 25% unentschieden endeten. Sofern nicht anders angegeben, gilt für alle Experimente δ = 16. Der Gradient wird über jeweils = 128 Quadrupel mittels Adam [KB15] und einer Lernrate α = 0.0001 über für e = 40 Epochen, berechnet. Der Koeffizient für die L_2-Gewichtsstrafe beträgt 10^{-6}.

5.3.3.2 Ähnlichkeitssuche

In diesem Experiment soll festgestellt werden, ob durch STEVE Repräsentationen berechnet werden können, bei denen ähnliche Teams im gelernten Vektorraum nah beieinander liegen.
Dazu werden fünf europäische Spitzenmannschaften ausgewählt und die Distanzen zu allen anderen Fußballmannschaften in deren Liga berechnet. Anschließend werden die fünf ähnlichsten Teams (kleinste Distanz) in Tabelle 5.3 dargestellt. Es gilt zu beachten, dass die Distanz zwischen den entsprechenden Gewinnerrepräsentationen der Mannschaften verwendet wird.
Wie erwartet, ist deutlich zu erkennen, dass Spitzenteams anderen Spitzenteams ähnlich sind. Das Team, das dem FC Barcelona am ähnlichsten ist, ist beispielsweise Real Madrid. Beide Mannschaften konkurrieren häufig um die Vorherrschaft in der spanischen La Liga. Selbiges gilt für den FC Bayern München und den RB Leipzig in der deutschen Bundesliga.
Generell lässt sich feststellen, dass die Ähnlichkeiten in Tabelle 5.3 in etwa die durchschnittliche Platzierung in der jeweiligen Liga widerspiegeln.

Ausgewählte Top-Fußballmannschaft pro Liga				
Bayern München	Barcelona	Paris SG	Manch. U.	Juve. Turin
Die fünf ähnlichsten Teams, ausgewählt von *STEVE*				
RB Leipzig	Real Madrid	Lyon	Liverpool	SSC Napoli
Dortmund	Valencia	Marseille	Manch. C.	AS Roma
Mönchengladb.	Atl. Madrid	Monaco	Chelsea	AC Milan
Leverkusen	Sevilla	St. Etienne	Tottenham	It. Milano
Hoffenheim	Villarreal	Lille	Arsenal	SS Lazio

Tabelle 5.3: Die fünf ähnlichsten Teams für fünf europäische Spitzenmannschaften.

5.3.3.3 Ranking von Fußballmannschaften

Um eine Rangliste von Fußballmannschaften zu erhalten, könnte im einfachsten Fall eine Ligatabelle verwendet werden. Allerdings spiegelt die Tabelle nur die Verfassung der Mannschaft in einer einzigen Saison wider. Frühere Erfolge werden in der Rangliste nicht berücksichtigt. Dieses Problem könnte umgangen werden, indem die Ligatabelle über mehrere Saisons hinweg gemittelt wird. Dabei ergibt sich jedoch ein weiteres Problem: Die Liste enthält nur Mannschaften aus einer einzigen Liga. Die Kombination von Tabellen aus verschiedenen Ländern und Wettbewerben, um eine vielfältigere Rangliste zu erhalten, ist wesentlich schwieriger zu bewerkstelligen. Noch komplizierter wird es, wenn eine Rangordnung für eine Liste eigens gewählter Mannschaften, möglicherweise aus vielen verschiedenen Ländern, erstellt werden soll.

STEVE bietet eine einfache, aber effektive Möglichkeit, Ranglisten für den oben genannten Anwendungsfall zu erstellen. Ausgehend von einer Liste an Mannschaften wird ein Turnier simuliert, bei dem jede Mannschaft gegen alle anderen Mannschaften spielt. Die Liste wird anschließend nach der Anzahl der Siege sortiert. Um das Ergebnis eines einzelnen Spiels (Sieg oder Niederlage) zwischen Mannschaft a und b zu berechnen, sei $\alpha = ||\Phi_a - \Psi_b||^2$ und $\beta = ||\Phi_b - \Psi_a||^2$. Wenn $\alpha < \beta$, dann liegt die Verlierer-Repräsentation von Team b näher an der Gewinnerrepräsentation von Team a als die Verlierer-Repräsentation von Team a an der Gewinnerrepräsentation von Team b. Somit ist Team a stärker als Team b und der Siegzähler von Team a wird um eins erhöht. Die gleichen Überlegungen gelten für den Fall $\alpha > \beta$.
In Abbildung 5.6 wurden zwei Ranglisten nach dem oben beschriebenen Verfahren erstellt. Jede Liste besteht aus zwölf Mannschaften aus verschiedenen europäischen Ländern mit unterschiedlicher Spielstärke. Die dargestellten Rangordnungen sind als sinnvoll zu erachten: Sehr erfolgreiche internationale Spitzenmannschaften wie Real Madrid, FC Bayern München, FC Barcelona und AS Roma stehen an der Spitze, während mittelmäßige Mannschaften wie Espanyol Barcelona und Werder Bremen die mittleren Plätze belegen. Die am wenigsten erfolgreichen Mannschaften wie FC Toulouse, Cardiff City, Fortuna Düsseldorf und Parma Calcio bilden das Schlusslicht der Listen. STEVE kann somit als eine Alternative zu früheren Fußballmannschafts-Ranking-Methoden [HA10; LZH10] gesehen werden, die auf dem ELO-Rating [Elo78] basieren.

[1] Real Madrid, FC Bayern München, Inter Milano, Liverpool FC, Borussia Dortmund, Ajax Amsterdam, FC Porto, Club Brugge KV, Werder Bremen, 1.FC Nürnberg, FC Toulouse, Cardiff City
[2] FC Barcelona, AS Roma, Atlético Madrid, Paris SG, Tottenham, PSV Eindhoven, Arsenal London, SL Benfica, Espanyol Barcelona, VFB Stuttgart, Fortuna Düsseldorf, Parma Calcio

Abbildung 5.6: Mit STEVE erstellte Mannschaftsranglisten. Jede Zeile[1,2] zeigt eine Rangliste vom stärksten (links) bis zum schwächsten Team (rechts). Die Zahlen über den Mannschaftslogos stehen für die relative Stärke einer Mannschaft, d. h. die Anzahl der gewonnenen hypothetischen Spiele.

5.3.3.4 Marktwert-Schätzung

Ziel von STEVE ist es auch, Repräsentationen zu erlernen, die für verschiedene nachgelagerte maschinelle Lernaufgaben gut geeignet sind. Ob STEVE diese

Anforderungen erfüllt, wird im folgenden Experiment im Hinblick auf seine Regressions- und Klassifikationsleistung evaluiert.
Das Experiment in diesem Abschnitt basiert auf einer einfachen Überlegung: Aussagekräftige Repräsentationen sollten genügend Informationen enthalten, um den Marktwert eines Teams zuverlässig zu schätzen. Der Marktwert einer Mannschaft enthält implizit auch Informationen über die Leistungsstärke einer Mannschaft. Insofern gilt es zu untersuchen, ob anhand der berechneten Repräsentationen der Marktwert einer Mannschaft vorhergesagt werden kann. Zu diesem Zweck wurden die Marktwerte für jedes Team im Datensatz aus der Saison 2018/2019 zusammengetragen, wobei sich der Marktwert einer Mannschaft aus der Summe der Marktwerte aller Spieler ergibt.
Im Durchschnitt ist ein Team 183,7 Millionen Euro (€M) wert, mit einer Standardabweichung von 241,2 Millionen Euro. Die geringwertigste Mannschaft ist BV De Graafschap (Niederlande, €10,15M) und die wertvollste Mannschaft ist der FC Barcelona (Spanien, €1180M). Das erste, zweite und dritte Quartil liegt bei €25M, €93,7M bzw. €232,5M.
Zur Marktwertschätzung ergeben sich zwei Problemstellungen, für welche ein mehrschichtiges FFNN verwendet wird:

Regression Die Schätzung von Mannschaftswerten lässt sich naturgemäß als Regressionsproblem darstellen, indem das FFNN die standardisierten Marktwerte vorhersagt. Die Standardisierung der Zielgröße (Marktwert) ist gängige Praxis im maschinellen Lernen, da nicht skalierte Zielvariablen zu explodierenden Gradienten führen können. Außerdem würden Zielvariablen im Millionenbereich bewirken, dass das FFNN hohe Gewichtswerte lernen muss, um diese aus den Repräsentationen vorherzusagen, was wiederum zu einem (numerisch) instabilen Trainingsprozess führt.
Die Auswertung erfolgt mittels 5-facher Kreuzvalidierung, die Ergebnisse sind in Tabelle 5.4 aufgeführt.

Klassifikation Durch die Gruppierung der Teamwerte in diskrete Klassen (sog. Bins) wird die Marktwertschätzung als Klassifizierungsproblem formuliert. Die Mannschaften werden entsprechend dem Quartil, in welchem ihr Wert liegt, in Klassen eingeteilt. Folglich wird jede Mannschaft einer von vier Klassen zugeordnet. Es werden das gleiche Standardisierungsverfahren und Evaluationsprotokoll wie bei der Regression verwendet. Die Ergebnisse sind in Tabelle 5.5 aufgeführt.

Das FFNN besitzt zwei versteckte Schichten der Größe 50 und 20. Abgesehen von der Änderung der Größe der versteckten Schichten werden für alle weiteren Analysen die von scikit-learn [Ped+11] bereitgestellten Standardparameter verwendet.
Für beide Problemstellungen werden sowohl durch STEVE erstellte Repräsentationen, als auch solche, die durch manuelles Feature Engineering erstellt wer-

den können, evaluiert. Der Vergleich beider Ansätze lässt damit Rückschlüsse darauf zu, ob STEVE den traditionelleren Verfahren überlegen ist.
Konkret werden folgende Repräsentationen verglichen:

i) STEVE: Die Repräsentationen werden mit STEVE mit $\delta \in \{8, 16, 32\}$ berechnet, wobei die Gewinner- und Verliererrepräsentationen konkateniert werden. Die Repräsentationen haben die Größe $16, 32$ und 64.

ii) Saisonstatistiken: Hier werden zählbasierte Merkmale für jedes Team im Datensatz extrahiert, um das manuelle Feature Engineering nachzuahmen. Für die Saison 2018/2019 wurden die folgenden Statistiken gesammelt: Anzahl der Siege, Unentschieden, Niederlagen sowie erzielte und kassierte Tore. Jedes Merkmal wird für Spiele in der Champions League, Euro League und der jeweiligen nationalen Liga berechnet. Zusätzlich werden Tore pro Spiel sowie Tore pro nationalem und internationalem Spiel verwendet. Daraus ergibt sich eine 18 dimensionale Repräsentation für jede Mannschaft.

iii) Saisonstatistiken (kat-x): Die Saisonstatistiken für die letzten x Saisons werden konkateniert. Die Repräsentationen haben die Größe $x * 18$.

iv) Saisonstatistiken (sum-x): Die Saisonstatistiken für die letzten x Saisons werden summiert. Die Merkmalsvektoren haben die Größe 18.

Die Vergleichbarkeit zwischen den verschiedenen oben genannten Repräsentationen ist dadurch gewährleistet, dass keine von ihnen Informationen verwendet, die im Datensatz fehlen.
Saisonstatistiken haben viele Eigenschaften, die intuitiv gut für die Schätzung von Mannschaftswerten geeignet sind. Zum Beispiel ist ein großer Teil der Mannschaften, die an internationalen Wettbewerben teilnehmen, wertvoller als die, welche das nicht tun. Statistiken über Tore und Spielergebnisse sind hilfreich, um die Stärke einer Mannschaft zu beurteilen, die wiederum mit dem Marktwert der Mannschaft korreliert.

Ergebnisse STEVE übertrifft die anderen Darstellungen sowohl in Bezug auf die Regressions- als auch die Klassifikationsleistung deutlich. Während $\delta = 64$ im Allgemeinen die besten Ergebnisse liefert, führt sogar $\delta = 16$ im Vergleich zu Saisonstatistiken zu besseren Ergebnissen. In Bezug auf die Regressionsleistung ist zu beobachten, dass Saisonstatistiken am konkurrenzfähigsten ist, wenn Informationen aus mehreren Saisons (kat-x und sum-x) verwendet werden. Allen Repräsentationen gelingt es, die allgemeine Tendenz des Marktwerts eines Teams abzuschätzen. Die Vorhersagen von STEVE sind jedoch weitaus präziser.
Ähnliche Schlussfolgerungen lassen sich bei der Untersuchung der Klassifizierungsleistung ziehen. Die beste konkurrierende Repräsentation ist Saisonstatistiken (kat-9), die 162-dimensional ist, 92 Dimensionen mehr als STEVE

	RMSE	MAE	MMAE
STEVE-16	142.12 ± 75.22	88.37 ± 25.69	52.01 ± 13.42
STEVE-32	131.51 ± 40.15	83.20 ± 24.51	46.87 ± 21.89
STEVE-64	**111.27** ± 48.58	**67.14** ± 30.51	**32.80** ± 10.42
Saisonstat.	173.75 ± 119.35	110.32 ± 63.61	69.96 ± 15.43
Saisonstat. (kat-3)	200.77 ± 157.55	138.15 ± 87.06	86.98 ± 39.83
Saisonstat. (kat-5)	172.05 ± 70.83	119.81 ± 43.18	80.74 ± 18.96
Saisonstat. (kat-9)	151.09 ± 80.37	105.98 ± 41.96	68.82 ± 23.15
Saisonstat. (sum-3)	158.44 ± 108.50	105.65 ± 53.95	69.16 ± 11.39
Saisonstat. (sum-5)	154.71 ± 115.76	104.04 ± 59.34	69.81 ± 15.69
Saisonstat. (sum-9)	158.33 ± 120.90	106.67 ± 62.61	67.74 ± 17.75

Tabelle 5.4: Ergebnisse für die Teamwertschätzung. Um die Qualität der Vorhersage zu quantifizieren, werden der mittlere quadratische Fehler (RMSE), der mittlere absoluten Fehler (MAE) und die mittlere Abweichung vom Median (MMAE) herangezogen. Die Werte sind angegeben in Millionen €.

	Micro F_1	Macro F_1
STEVE-16	0.67 ± 0.10	0.64 ± 0.10
STEVE-32	**0.74** ± 0.11	0.71 ± 0.14
STEVE-64	**0.74** ± 0.10	**0.72** ± 0.09
Saisonstat.	0.52 ± 0.14	0.45 ± 0.19
Saisonstat. (kat-3)	0.50 ± 0.12	0.44 ± 0.15
Saisonstat. (kat-5)	0.55 ± 0.14	0.51 ± 0.16
Saisonstat. (kat-9)	0.60 ± 0.13	0.56 ± 0.11
Saisonstat. (sum-3)	0.49 ± 0.09	0.40 ± 0.10
Saisonstat. (sum-5)	0.48 ± 0.08	0.39 ± 0.07
Saisonstat. (sum-9)	0.47 ± 0.09	0.37 ± 0.15

Tabelle 5.5: Ergebnisse für die Klassifizierungsaufgabe der Marktwertschätzung, gemessen mit den Metriken Micro F_1 und Macro F_1.

(δ = 64). Dennoch bietet STEVE (δ = 64) eine $\approx$ 20% bessere Leistung als Saisonstatistiken (kat-9). Daraus lässt sich schließen, dass STEVE in der Lage ist, die für die Aufgabe benötigten Informationen zu erfassen, und es STEVE gelingt, aussagekräftige Repräsentationen zu berechnen.

5.3.4 Fazit und Verbesserungsmöglichkeiten

In dieser Arbeit wurde STEVE vorgestellt, eine einfache, aber effektive Methode, um Repräsentationen für Fußballmannschaften zu berechnen.
Eine qualitative Analyse ergab die Eignung der Repräsentationen für das Team-Ranking und die Ähnlichkeitssuche. Quantitativ konnte die Nützlichkeit des Ansatzes für die Schätzung von Marktwerten von Fußballmannschaften bestätigt werden. In beiden Fällen gelingt es STEVE, sinnvolle und effektive Darstellungen zu liefern. Zukünftige Arbeiten könnten verschiedene Gewichtungsschemata für die Saison, in der ein Spiel stattgefunden hat, weiter untersuchen. Zum Beispiel kann anstelle der linearen Gewichtung die Exponentialverteilung verwendet werden. Darüber hinaus könnte die Einbeziehung der Anzahl der während eines Spiels erzielten Tore und die Berücksichtigung des Heimvorteils helfen, weitere Feinheiten zu erfassen.

5.4 Kapitelzusammenfassung

Dieses Kapitel behandelte Anwendungen des Repräsentationslernens, welche über die Verarbeitung akustischer Daten hinausgehen.
Zunächst wurde mit ClusterComm (Abschnitt 5.1) eine neue Methode zur effizienten und diskreten Kommunikation im Mulit-Agent Reinforcement Learning vorgeschlagen. ClusterComm nutzt als Nachricht die Clusterindizes, welche durch das Clustering der internen Repräsentationen der Agenten entstehen. Durch die Evaluation in vier verschiedenen Evaluationsumgebungen wurde gezeigt, dass ClusterComm mit der Leistung eines unbegrenzten und kontinuierlichen Kommunikationskanals konkurrieren kann.
Überdies wurde die Problemstellung der Anomalieerkennung im Reinforcement Learning konkretisiert und formalisiert (Abschnitt 5.2). Zunächst wird angenommen, dass der Zustandsraum und die Übergangsdynamik durch eine Kontextvariable gesteuert werden. Auf Basis dieser Annahme wurde das Problem dann darauf zurückgeführt zu erkennen, ob der Agent sich in einem bekannten oder unbekannten Kontext befindet.
Zuletzt (Abschnitt 5.3) wurde STEVE vorgestellt, ein Algorithmus, welcher das Lernen von Repräsentationen von Fußballmannschaften erlaubt. STEVE benötigt ausschließlich frei verfügbare Daten von Spielergebnissen. Zudem konnte empirisch belegt werden, dass sich STEVE besser für nachgelagerte Lernaufgaben eignet als Ansätze, die auf dem manuellen Feature Engineering basieren.

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit stand das Repräsentations- und Transferlernen für Anwendungen des maschinellen Lernens im Mittelpunkt. Die vorgestellten Methoden und Algorithmen sind dabei allesamt dem Deep Learning zuzuordnen. Das Deep Learning umfasst die Verwendung von neuronalen Netzen für maschinelle Lernprobleme. Neuronale Netze stellen dabei die aktuell flexibelste und leistungsfähigste Modellklasse dar. Durch deren Verwendung konnten in jüngster Zeit beachtliche Fortschritte in fast allen Bereichen der Wissenschaften erzielt werden. In dieser Arbeit wurde zudem besonderer Wert auf den praktischen Nutzen der entwickelten Verfahren gelegt. Die untersuchten Anwendungsszenarien umfassen unter anderem die Klassifikation und Anomalieerkennung akustischer Daten. Überdies wurden Methoden zur Absicherung und Kommunikation in Multiagentensystemen vorgestellt. Darüber hinaus konnte gezeigt werden, wie das Repräsentationslernen selbst im Mannschaftssport einen Mehrwert bieten kann. Allgemein lässt sich somit festhalten, dass die vorliegende Arbeit ein umfangreiches und vielfältiges Anwendungsspektrum des maschinellen Lernens abdeckt. Die Arbeit umfasst zahlreiche Experimente, neu entwickelte Algorithmen und empirische Vergleiche von bestehenden Ansätzen für neuartige Anwendungsbereiche. Dadurch konnte die besondere Bedeutung des Repräsentations- und Transferlernens für das angewandte maschinelle Lernen unterstrichen werden.
Im Folgenden werden nochmals die wichtigsten Erkenntnisse und Ergebnisse der einzelnen Kapitel zusammengefasst.
Kapitel 3 widmete sich zunächst dem Repräsentationslernen auf Basis akustischer Daten. Dazu wurde in Abschnitt 3.1 eine rekurrente neuronale Netzarchitektur zur Klassifikation von Primatenvokalisationen vorgestellt. Die Architektur ist inspiriert vom manuellen Feature Engineering, bei dem Merkmalsvektoren für kleine Ausschnitte des Mel-Spektrogramms extrahiert werden. Der vorgestellte Ansatz verbessert dieses Verfahren, indem er die sequentielle Natur akustischer Daten berücksichtigt und die Merkmalsextraktion komplett differenzierbar macht. Somit können Repräsentationen automatisch und datengetrieben aus dem zugrundeliegenden Mel-Spektrogramm gewonnen werden. Die Aggregation der Repräsentationen erfolgt dann mit Hilfe einfacher Pooling-Mechanismen. Die besondere Eignung der Architektur für die Wildtierüberwachung konnte dadurch bestätigt werden, dass ihre Genauigkeit vergleichbare Ansätze deutlich übertrifft.

Ein weiterer Schwerpunkt des Kapitels 3 behandelte die akustische Anomalieerkennung. Dazu wurde in Abschnitt 3.2 zunächst ein generelles Verfahren zur Leckerkennung in Wasserversorgungsnetzen vorgestellt. Zu diesem Zwecke wurden in einem Versuchsgebiet Körperschallmikrofone entlang des Wasserversorgungsnetzwerks angebracht. Anschließend wurde ein Datensatz mit akustischen Aufnahmen des Normalbetriebs sowie Aufnahmen von Leckgeräuschen erstellt. Über einen zuvor festgelegten Entscheidungshorizont wurden periodisch akustische Aufnahmen entnommen und mit Hilfe verschiedener Anomalieerkennungmodelle bewertet. Dazu konnte zunächst festgestellt werden, dass die Erkennung von Lecks, die nah an einem Körperschallmikrofon auftreten, trivial ist. Bezüglich weiter entfernter Lecks zeigte sich jedoch, dass neuronale Netze eine bessere Erkennungsleistung aufweisen. Zudem wurde untersucht, wie häufig Aufnahmen innerhalb des Entscheidungshorizonts entnommen werden müssen. Die Ergebnisse zeigen klar, dass nur ein Bruchteil der Gesamtdauer des Entscheidungshorizonts berücksichtigt werden muss. So lassen sich im angedachten Anwendungsszenario Übertragungsbandbreite und Energie sparen.
In Abschnitt 3.3 wurde DRINK, ein neuer Algorithmus zur Erkennung von akustischen Anomalien in Aufnahmen von Industriemaschinen, vorgestellt. DRINK entfernt zunächst eine bestimmte Anzahl der mittleren Zeitfenster des Mel-Spektrogramms. Die verbleibenden Zeitfenster werden anschließend an ein rekurrentes neuronales Netz übergeben, um die zuvor entfernen Zeitfenster vorherzusagen. Der Vorhersagefehler lässt sich dann zur Anomaliebewertung einsetzen. Durch dieses Vorgehen wird das Problem des konstant hohen Rekonstruktionsfehlers der außenliegenden Zeitfenster umgangen. Die Hypothese, dass Normalfalldaten dadurch deutlich besser rekonstruiert werden können als es bei einem klassischen Autoencoder der Fall wäre und dass der Rekonstruktionsfehler für Anomalien trotzdem hoch bliebe, ließ sich experimentell bestätigen. In den meisten Fällen weist DRINK eine bessere Erkennungsleitung als vergleichbare Ansätze auf. Vor allem bei nicht stationären Geräuschen ließ sich feststellen, dass Kontextinformationen für eine gute Vorhersage besonders wichtig sind.
Im darauffolgenden Kapitel 4 wurden anschließend die Auswirkungen des Wissenstransfers von vortrainierten neuronalen Netzen zur akustischen Anomalieerkennung untersucht. Der Vorteil dieses Vorgehens ist, dass bereits trainierte und frei verfügbare neuronale Netze zur Merkmalsextraktion verwendet werden können, ohne dass sie von Grund auf neu entwickelt und trainiert werden müssen. Abschnitt 4.1 untersucht dazu die Tauglichkeit von Convolutional Neural Networks, die zur Bildklassifizierung vortrainiert wurden. Es konnte gezeigt werden, dass ResNet [He+16] Architekturen in Kombination mit Gaussian-Mixture-Models und One-Class Support Vector Machines [Sch+99] bessere Ergebnisse erzielen als ein Convolutional Autoencoder.
Aufbauend auf diesen Ergebnissen, wurden in Abschnitt 4.2 für den gleichen Anwendungsfall auch neuronale Netze untersucht, die auf Geräusch- und Mu-

sikaufnahmen vortrainiert wurden. Die Ergebnisse legen eine deutliche Überlegenheit der musikbasierten Repräsentationen nahe, obwohl Musik große strukturelle Unterschiede zu Maschinengeräuschen aufweist. Auch im Vergleich zu DRINK konnte die Konkurrenzfähigkeit des Ansatzes nachgewiesen werden.
Im letzten Kapitel 5 wurden schließlich Anwendungen des Repräsentationslernens besprochen, die über die Verarbeitung akustischer Daten hinausgehen. In Abschnitt 5.1 wurde ClusterComm vorgestellt, ein neuartiges Verfahren zur Ermöglichung von diskreter Kommunikation im Multi-Agent Reinforcement Learning. ClusterComm diskretisiert die internen Repräsentationen der eingehenden Beobachtung eines jeden Agenten und verwendet die Clusterindizes als Nachricht. Im Gegensatz zu den meisten anderen Ansätzen lernen die ClusterComm-Agenten unabhängig und ohne die Verwendung von zentralisierten Lernmethoden. In dem Abschnitt konnte gezeigt werden, dass ClusterComm alle Ansätze ohne Kommunikation übertrifft. Darüber hinaus konkurriert ClusterComm mit der Leistung eines uneingeschränkten und kontinuierlichen Kommunikationskanals.
Im anschließenden Abschnitt 5.2 wurde erneut die Problemstellung der Anomalieerkennung aufgegriffen, hier jedoch in Bezug auf die Absicherung von Reinforcement Learning Systemen. Da das Problem in der Literatur zum aktuellen Zeitpunkt nicht ausreichend konkretisiert und formalisiert ist, wurde es in diesem Abschnitt präzisiert. Dazu wurden der aktuelle Stand der Forschung zusammengefasst und offene Probleme herausgearbeitet. Anschließend wurde versucht, das Problem anhand von Kontextrepräsentationen, die den Zustandsraum und die Übergangsdynamik der Umgebung steuern, zu formalisieren. Zusätzlich wurden praktische Desiderata für zukünftige Probleme formuliert.
Im letzten Abschnitt 5.3 wurde STEVE vorgestellt, ein Algorithmus zum Lernen von Repräsentationen von Fußballmannschaften. STEVE benötigt lediglich Zugriff auf frei verfügbare Spielergebnisse. Im gelernten Merkmalsraum befinden sich Repräsentationen von ähnlichen Mannschaften in direkter Nachbarschaft. STEVE verringert die Abhängigkeit von teureren Fachexperten und Datensätzen und kann genutzt werden, um Mannschaften zu vergleichen oder um Spielergebnisse und den Marktwert vorherzusagen. Experimentell konnte belegt werden, dass sich STEVE dazu besser eignet als das manuelle Feature Engineering.
Am Ende jedes Abschnitts wurden bereits Verbesserungsvorschläge und Ansatzpunkte für zukünftige Arbeiten dargelegt. An dieser Stelle soll deshalb auf offene Fragestellungen und mögliche Lösungsansätze eingegangen werden, die auf einen möglichst großen Teil der in dieser Arbeit vorgestellten Ansätze zutreffen.
Alle diese Ansätze verwenden entweder faltende- oder rekurrente neuronale Netze. Zukünftige Arbeiten könnten auf Basis von Transformern [Vas+17] entwickelt werden. Transformer haben sich in letzter Zeit durch die Verwendung des Self-Attention Mechanismus als leistungsstarke Alternative erwiesen. Sie

wurden zunächst für die Verarbeitung von Sequenzen entwickelt, sind aber mittlerweile auch in der Computervision [Dos+21] und der akustischen Signalverarbeitung [Gon+22] allgegenwärtig. Da sowohl Mel-Spektrogramme als auch die Beobachtungen und Nachrichten von Agenten als Sequenz aufgefasst werden können, stellt die Anwendung von Transformern eine sinnvolle Alternative dar. Auch die Evaluation von bereits vortrainierten Transformern ist denkbar.
Ein zweiter Punkt, den es zu verbessern gilt, ist die Interpretierbarkeit [Zha+21b] der Modelle. In ihrer aktuellen Form ist es schwer, die Vorhersagen und Entscheidungen der vorgestellten Ansätze zu erklären. Es wäre also wünschenswert, dass die Modelle zusätzlich zur Vorhersage auch eine Beschreibung ausgeben, wie sie zu der Vorhersage gekommen sind. Im Falle der Anomalieerkennung ließe sich so ablesen, welche Faktoren den Normalfall beschreiben und in welchen Situationen das Modell die Eingabedaten als anormal einstuft. Auch im Reinforcement Learning ließen sich so die gelernten Handlungsstrategien besser nachvollziehen. Darüber hinaus kann eine bessere Interpretierbarkeit die Entwicklung von neuen und besseren Modellen beschleunigen, da Fehler, die das Modell macht, schneller aufgedeckt werden könnten.
Bezüglich des Transferlernens ist anzumerken, dass es einer aufwändige Evaluation bedarf, um festzustellen, welche Quelldomäne für eine bestimmte Zielaufgabe geeignet ist. Da sich die Evaluation auf annotierte Daten stützt und rechenintensiv ist, sollten Methoden entwickelt werden, die dieses Problem vermeiden. Aktuelle Ansätze [AF20; Pan+22] quantifizieren die Übertragbarkeit zwischen einem Ausgangsmodell und einem Zieldatensatz durch die Analyse des induzierten Merkmalraums.
Zuletzt ist noch die Verwendung weiterer Ansätze des Self-Supervised Learnings angebracht. Abgesehen von Autoencodern, welche die Eingabe rekonstruieren, existieren noch weitere Ansätze, die zum Lernen keine Labels benötigen. Diese [Che+20; He+20; Gri+20; Bae+22] versuchen die Invarianz der Repräsentation gegenüber bestimmter Transformationen (z.B. Änderung der Farbe, Spiegeln, Ausschnitte entfernen) herzustellen. Diese Problemformulierung unterscheidet sich also konzeptionell deutlich von der des Autoencoders. Es ist folglich noch zu untersuchen, ob solche Repräsentationen für die in dieser Arbeit betrachteten Anwendungsszenarien einen Mehrwert bieten können. Die angesprochenen Fragestellungen wurden in der Literatur bisher noch nicht weiter untersucht und bilden somit vielversprechende Anknüpfungspunkte für Folgearbeiten auf dem Gebiet des Repräsentations- und Transferlernens.

Abkürzungsverzeichnis

AE Autoencoder

CNN Convolutional Neural Network

dB Decibel

DRINK Deep Recurrent Interpolation Network

FFNN Feedforward Neural Network

GMM Gaussian-Mixture Model

GRU Gated recurrent Unit

Hz Hertz

KL-Divergenz Kullback-Leibler Divergenz

LSTM Long Short-Term Memory

MARL Multi-Agent Reinforcement Learning

MBRL Model based Reinforcement Learning

NN Neuronales Netz

RL Reinforcement Learning

RNN Recurrent Neural Network

STEVE Soccer Team Vectors

OC-SVM One-Class Support Vector Machine

Literatur

[Ade+17] Kazeem B Adedeji, Yskandar Hamam, Bolanle Tolulope Abe und Adnan M Abu-Mahfouz. „Towards achieving a reliable leakage detection and localization algorithm for application in water piping networks: An overview“. In: *IEEE Access* 5 (2017), S. 20272–20285. DOI: 10.1109/ACCESS.2017.2752802.

[AF20] David Alvarez-Melis und Nicolò Fusi. „Geometric Dataset Distances via Optimal Transport“. In: NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[Aga18] Abien Fred Agarap. „Deep learning using rectified linear units (relu)“. In: *arXiv preprint arXiv:1803.08375* (2018).

[AHK01] Charu C Aggarwal, Alexander Hinneburg und Daniel A Keim. „On the surprising behavior of distance metrics in high dimensional space“. In: *International conference on database theory*. Springer. 2001, S. 420–434. DOI: 10.1007/3-540-44503-X_27.

[Aki+19] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta und Masanori Koyama. „Optuna: A Next-Generation Hyperparameter Optimization Framework“. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, S. 2623–2631. DOI: 10.1145/3292500.3330701.

[Alc16] Michael A Alcorn. „(batter| pitcher)2vec: statistic-free talent modeling with neural player embeddings“. In: MIT Sloan Sports Analytics Conference. 2016.

[AM10] J Arbonés und P Milrud. *Die Mathematik der Musik*. 2010.

[Ami+17] Shahin Amiriparian, Maurice Gerczuk, Sandra Ottl, Nicholas Cummins, Michael Freitag, Sergey Pugachevskiy, Alice Baird und Björn Schuller. „Snore Sound Classification Using Image-Based Deep Spectrum Features“. In: *Proc. Interspeech 2017*. 2017, S. 3512–3516. DOI: 10.21437/Interspeech.2017-434.

[Ami+20] Shahin Amiriparian, Nicholas Cummins, Maurice Gerczuk, Sergey Pugachevskiy, Sandra Ottl und Björn Schuller. „Äre You Playing a Shooter Again?!"Deep Representation Learning for Audio-Based Video Game Genre Recognition". In: *IEEE Transactions on Games* 12.2 (2020), S. 145–154. DOI: 10.1109/TG.2019.2894532.

[Amo+16] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman und Dan Mané. „Concrete problems in AI safety". In: *arXiv:1606.06565* (2016).

[Ayt+18] Caglar Aytekin, Xingyang Ni, Francesco Cricri und Emre Aksu. „Clustering and unsupervised anomaly detection with l2 normalized deep auto-encoder representations". In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2018, S. 1–6. DOI: 10.1109/IJCNN.2018.8489068.

[AZ17] Relja Arandjelovic und Andrew Zisserman. „Look, Listen and Learn". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, S. 609–617. DOI: 10.1109/ICCV.2017.73.

[Bae+22] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu und Michael Auli. „data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language". In: *Proceedings of the 39th International Conference on Machine Learning*. Bd. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, S. 1298–1312.

[Bal+21] Philip Ball, Cong Lu, Jack Parker-Holder und S Roberts. „Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment". In: *Self-Supervision for Reinforcement Learning Workshop-ICLR 2021*. 2021.

[BB86] Ronald Newbold Bracewell und Ronald N Bracewell. *The Fourier transform and its applications*. Bd. 31999. McGraw-Hill New York, 1986. DOI: 10.1119/1.1973431.

[BCB15] Dzmitry Bahdanau, Kyung Hyun Cho und Yoshua Bengio. „Neural machine translation by jointly learning to align and translate". In: *3rd International Conference on Learning Representations (ICLR)*. 2015.

[BD13] Michael Bittle und Alec Duncan. „A review of current marine mammal detection and classification algorithms for use in automated passive acoustic monitoring". In: *Annual Conference of the Australian Acoustical Society 2013, Acoustics 2013: Science, Technology and Amenity* (Jan. 2013), S. 208–215.

[BDI21] Barış Bayram, Taha Berkay Duman und Gökhan Ince. „Real time detection of acoustic anomalies in industrial processes using sequential autoencoders“. In: *Expert Systems* 38.1 (2021). DOI: 10.1111/exsy.12564.

[Bec+19] Pierre Beckmann, Mikolaj Kegler, Hugues Saltini und Milos Cernak. „Speech-VGG: A deep feature extractor for speech processing“. In: *arXiv preprint arXiv:1910.09909* (2019).

[Beg+23] Karim Beguir u. a. „Early computational detection of potential high-risk SARS-CoV-2 variants“. In: *Computers in Biology and Medicine* 155 (2023), S. 106618. DOI: 10.1016/j.compbiomed.2023.106618.

[Bej+17] Babak Ehteshami Bejnordi u. a. „Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer“. In: *Jama* 318.22 (2017), S. 2199–2210. DOI: 10.1001/jama.2017.14585.

[Bel54] Richard Bellman. „The theory of dynamic programming“. In: *Bulletin of the American Mathematical Society* 60.6 (1954), S. 503–515. DOI: 10.1073/pnas.38.8.716.

[Ben+03] Yoshua Bengio, Réjean Ducharme, Pascal Vincent und Christian Jauvin. „A neural probabilistic language model“. In: *Journal of machine learning research* 3.Feb (2003), S. 1137–1155. DOI: 10.1007/3-540-33486-6_6.

[Ber+11] James Bergstra, Rémi Bardenet, Yoshua Bengio und Balázs Kégl. „Algorithms for Hyper-Parameter Optimization“. In: *Advances in Neural Information Processing Systems*. Bd. 24. Curran Associates, Inc., 2011.

[Bij+18] Maryam Bijanzadeh, Lauri Nurminen, Sam Merlin, Andrew M Clark und Alessandra Angelucci. „Distinct laminar processing of local and global context in primate primary visual cortex“. In: *Neuron* 100.1 (2018), S. 259–274. DOI: 10.1016/j.neuron.2018.08.020.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 2006. DOI: 10.1117/1.2819119.

[BK19] Steven L. Brunton und J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019. DOI: 10.1017/9781108380690.

[BKM17] David M Blei, Alp Kucukelbir und Jon D McAuliffe. „Variational inference: A review for statisticians“. In: *Journal of the American statistical Association* 112.518 (2017), S. 859–877. DOI: 10.1080/01621459.2017.1285773.

[BLC13] Yoshua Bengio, Nicholas Léonard und Aaron Courville. „Estimating or propagating gradients through stochastic neurons for conditional computation“. In: *arXiv preprint arXiv:1308.3432* (2013).

[Blu+11] Daniel T Blumstein u. a. „Acoustic monitoring in terrestrial environments using microphone arrays: applications, technological considerations and prospectus“. In: *Journal of Applied Ecology* 48.3 (2011), S. 758–767. DOI: 10.1111/j.1365-2664.2011.01993.x.

[BPP20] Emmanuel Bengio, Joelle Pineau und Doina Precup. „Interference and Generalization in Temporal Difference Learning“. In: *Proceedings of the 37th International Conference on Machine Learning*. Bd. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, S. 767–777.

[Bra+16] Corinne L Brauer, Therese M Donovan, Ruth M Mickey, Jonathan Katz und Brian R Mitchell. „A comparison of acoustic monitoring methods for common anurans of the northeastern United States“. In: *Wildlife Society Bulletin* 40.1 (2016), S. 140–149. DOI: 0.1002/wsb.619.

[Bro+20] Tom Brown u. a. „Language Models are Few-Shot Learners“. In: *Advances in Neural Information Processing Systems*. Bd. 33. Curran Associates, Inc., 2020, S. 1877–1901.

[BT52] Ralph Allan Bradley und Milton E Terry. „Rank analysis of incomplete block designs: I. The method of paired comparisons“. In: *Biometrika* 39.3/4 (1952), S. 324–345. DOI: /10.2307/2334029.

[Bur+18] Yuri Burda, Harrison Edwards, Amos Storkey und Oleg Klimov. „Exploration by random network distillation“. In: *International Conference on Learning Representations (ICLR)*. 2018.

[Car+19] Mathilde Caron, Piotr Bojanowski, Julien Mairal und Armand Joulin. „Unsupervised Pre-Training of Image Features on Non-Curated Data“. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, S. 2959–2968. DOI: 10.1109/ICCV.2019.00305.

[CCM19] Dena J Clink, Margaret C Crofoot und Andrew J Marshall. „Application of a semi-automated vocal fingerprinting approach to monitor Bornean gibbon females in an experimentally fragmented landscape in Sabah, Malaysia“. In: *Bioacoustics* 28.3 (2019), S. 193–209. DOI: 10.1080/09524622.2018.1426042.

[CCZ18] TK Chan, Cheng Siong Chin und Xionghu Zhong. „Review of current technologies and proposed intelligent methodologies for water distributed network leakage detection“. In: *IEEE Access* 6 (2018), S. 78846–78867. DOI: 10.1109/ACCESS.2018.2885444.

[CF13] Anthony Constantinou und Norman Fenton. „Determining the level of ability of football teams by dynamic ratings based on the relative discrepancies in scores between adversaries“. In: *Journal of Quantitative Analysis in Sports* 9 (Jan. 2013), S. 37–50. DOI: 10.1515/jqas-2012-0036.

[Che+20] Ting Chen, Simon Kornblith, Mohammad Norouzi und Geoffrey Hinton. „A Simple Framework for Contrastive Learning of Visual Representations“. In: *Proceedings of the 37th International Conference on Machine Learning*. Bd. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, S. 1597–1607.

[Che+22] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick und Shlomo Dubnov. „HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection“. In: *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, S. 646–650. DOI: 10.1109/ICASSP43922.2022.9746312.

[Cho+14] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau und Yoshua Bengio. „On the Properties of Neural Machine Translation: Encoder–Decoder Approaches“. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, Okt. 2014, S. 103–111. DOI: 10.3115/v1/W14-4012.

[CJ16] Shuo Chen und Thorsten Joachims. „Predicting Matchups and Preferences in Context“. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, S. 775–784. DOI: 10.1145/2939672.2939764.

[CJM22] Guendalina Caldarini, Sardar Jaf und Kenneth McGarry. „A literature survey of recent advances in chatbots“. In: *Information* 13.1 (2022), S. 41. DOI: 10.3390/info13010041.

[CLF18] Edward Choi, Angeliki Lazaridou und Nando de Freitas. „Compositional Obverter Communication Learning from Raw Visual Input“. In: *International Conference on Learning Representations (ICLR)*. 2018.

[CNT17] Roya Cody, Sriram Narasimhan und Bryan Tolson. „One Class SVM – Leak Detection in Water Distribution Systems“. In: *Computing and Control for the Water Industry (CCWI)* (2017). DOI: 10.15131/shef.data.5363959.v1.

[Cob+20] Karl Cobbe, Chris Hesse, Jacob Hilton und John Schulman. „Leveraging Procedural Generation to Benchmark Reinforcement Learning“. In: *Proceedings of the 37th International Conference on Machine Learning*. Hrsg. von Hal Daumé III und Aarti Singh. Bd. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, S. 2048–2056.

[Cra+15] Michael Crawford, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter und Hamzah Al Najada. „Survey of review spam detection using machine learning techniques“. In: *Journal of Big Data* 2.1 (2015), S. 1–24. DOI: 10.1186/s40537-015-0029-9.

[Cra+19] Aurora Linh Cramer, Ho-Hsiang Wu, Justin Salamon und Juan Pablo Bello. „Look, Listen, and Learn More: Design Choices for Deep Audio Embeddings“. In: *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, S. 3852–3856. DOI: 10.1109/ICASSP.2019.8682475.

[CSA20] Filippos Christianos, Lukas Schäfer und Stefano Albrecht. „Shared Experience Actor-Critic for Multi-Agent Reinforcement Learning“. In: *Advances in Neural Information Processing Systems*. Hrsg. von H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan und H. Lin. Bd. 33. Curran Associates, Inc., 2020, S. 10707–10717.

[CTO20] Roya A Cody, Bryan A Tolson und Jeff Orchard. „Detecting Leaks in Water Distribution Pipes Using a Deep Autoencoder and Hydroacoustic Spectrograms“. In: *Journal of Computing in Civil Engineering* 34.2 (2020). DOI: 0.1061/(ASCE)CP.1943-5487.0000881.

[CTW19] Wei-Yi Chuang, Yao-Long Tsai und Li-Hua Wang. „Leak Detection in Water Distribution Pipes Based on CNN with Mel Frequency Cepstral Coefficients“. In: *Proceedings of the 2019 3rd International Conference on Innovation in Artificial Intelligence*. ICIAI 2019. New York, NY, USA: Association for Computing Machinery, 2019, S. 83–86. DOI: 10.1145/3319921.3319926.

[CUH16] Djork-Arné Clevert, Thomas Unterthiner und Sepp Hochreiter. „Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)“. In: *4th International Conference on Learning Representations (ICLR), ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. 2016.

[Cum+17] Nicholas Cummins, Shahin Amiriparian, Gerhard Hagerer, Anton Batliner, Stefan Steidl und Björn W Schuller. „An image-based deep spectrum feature representation for the recognition of emotional speech“. In: *Proceedings of the 25th ACM international conference on Multimedia*. 2017, S. 478–484. DOI: 10.1145/3123266.3123371.

[Dai+16] Jia Dai, Shan Liang, Wei Xue, Chongjia Ni und Wenju Liu. „Long short-term memory recurrent neural network based segment features for music genre classification“. In: *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE. 2016, S. 1–5. DOI: 10.1109/ISCSLP.2016.7918369.

[Das+19] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat und Joelle Pineau. „TarMAC: Targeted Multi-Agent Communication“. In: *Proceedings of the 36th International Conference on Machine Learning*. Bd. 97. Proceedings of Machine Learning Research. PMLR, Sep. 2019, S. 1538–1546.

[DBİ19] Taha Berkay Duman, Barış Bayram und Gökhan İnce. „Acoustic Anomaly Detection Using Convolutional Autoencoders in Industrial Processes“. In: *International Workshop on Soft Computing Models in Industrial and Environmental Applications*. Springer. 2019, S. 432–442. DOI: 10.1007/978-3-030-20055-8_41.

[DCS17] Yuxiao Dong, Nitesh V. Chawla und Ananthram Swami. „Metapath2vec: Scalable Representation Learning for Heterogeneous Networks“. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '17. New York, NY, USA: Association for Computing Machinery, 2017, S. 135–144. DOI: 10.1145/3097983.3098036.

[Dee+21] Lucas Deecke, Lukas Ruff, Robert A. Vandermeulen und Hakan Bilen. „Transfer-Based Semantic Anomaly Detection“. In: *Proceedings of the 38th International Conference on Machine Learning*. Bd. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, S. 2546–2558.

[Dem+19] Fatih Demir, Abdulkadir Şengür, Varun Bajaj und Kemal Polat. „Towards the classification of heart sounds based on convolutional deep neural network“. In: *Health information science and systems* 7.1 (2019), S. 1–9. DOI: 10.1007/s13755-019-0078-0.

[Den+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li und Li Fei-Fei. „Imagenet: A large-scale hierarchical image database“. In: *2009 IEEE conference on computer vision and pattern recognition*. IEEE. 2009, S. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[Dev+19] Jacob Devlin, Ming-Wei Chang, Kenton Lee und Kristina Toutanova. „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*. 2019, S. 4171–4186. DOI: 10.18653/v1/n19-1423.

[DF21] Mohamad H Danesh und Alan Fern. „Out-of-Distribution Dynamics Detection: RL-Relevant Benchmarks and Results“. In: *International Conference on Machine Learning, Uncertainty & Robustness in Deep Learning Workshop*. 2021.

[Dha+20] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford und Ilya Sutskever. „Jukebox: A Generative Model for Music“. In: *arXiv preprint arXiv:2005.00341* (2020).

[Dos+21] Alexey Dosovitskiy u. a. „An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale“. In: *The 9th International Conference on Learning Representations (ICLR)*. 2021.

[DSB17] Laurent Dinh, Jascha Sohl-Dickstein und Samy Bengio. „Density estimation using Real NVP“. In: *5th International Conference on Learning Representations (ICLR)*. 2017.

[Dun73] J. C. Dunn. „A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters“. In: *Journal of Cybernetics* 3.3 (1973), S. 32–57. DOI: 10.1080/01969727308546046.

[DVN20] Gauraang Dhamankar, Jose R. Vazquez-Canteli und Zoltan Nagy. „Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms on a Building Energy Demand Coordination Task“. In: *Proceedings of the 1st International Workshop on Reinforcement Learning for Energy Management in Buildings & Cities*. RLEM'20. New York, NY, USA: Association for Computing Machinery, 2020, S. 15–19. DOI: 10.1145/3427773.3427870.

[Ecc+19] Tom Eccles, Yoram Bachrach, Guy Lever, Angeliki Lazaridou und Thore Graepel. „Biases for emergent communication in multi-agent reinforcement learning“. In: *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[EL22] Benjamin Eysenbach und Sergey Levine. „Maximum Entropy RL (Provably) Solves Some Robust RL Problems“. In: *International Conference on Learning Representations*. 2022.

[Elo78] Arpad E Elo. *The rating of chessplayers, past and present*. Arco Pub., 1978.

[EWS10] Florian Eyben, Martin Wöllmer und Björn Schuller. „Opensmile: The Munich Versatile and Fast Open-Source Audio Feature Extractor“. In: *Proceedings of the 18th ACM International Conference on Multimedia*. MM '10. New York, NY, USA: Association for Computing Machinery, 2010, S. 1459–1462. DOI: 10.1145/1873951.1874246.

[EZ19] Samer El-Zahab und Tarek Zayed. „Leak detection in water distribution networks: an introductory overview“. In: *Smart Water* 4.1 (2019), S. 5. DOI: 10.1186/s40713-019-0017-x.

[Fit10] W. Tecumseh Fitch. *The Evolution of Language*. en. Cambridge University Press, Apr. 2010.

[Foe+16] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas und Shimon Whiteson. „Learning to Communicate with Deep Multi-Agent Reinforcement Learning“. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, S. 2145–2153.

[Fon+21] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font und Xavier Serra. „FSD50K: An Open Dataset of Human-Labeled Sound Events“. In: *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 30 (Dez. 2021), S. 829–852. DOI: 10.1109/TASLP.2021.3133208.

[For+20] Pierre Foret, Ariel Kleiner, Hossein Mobahi und Behnam Neyshabur. „Sharpness-aware Minimization for Efficiently Improving Generalization“. In: *International Conference on Learning Representations (ICLR)*. 2020.

[Fre+17] Michael Freitag, Shahin Amiriparian, Sergey Pugachevskiy, Nicholas Cummins und Björn Schuller. „AuDeep: Unsupervised Learning of Representations from Audio with Deep Recurrent Neural Networks“. In: *J. Mach. Learn. Res.* 18.1 (Jan. 2017), S. 6340–6344.

[FZD16] Pawel Fedurek, Klaus Zuberbühler und Christoph D Dahl. „Sequential information in a great ape utterance“. In: *Scientific Reports* 6.1 (2016), S. 1–11. DOI: 10.1038/srep38226.

[Gao+17] Yan Gao, Michael J Brennan, Yuyou Liu, Fabricio CL Almeida und Phillip F Joseph. „Improving the shape of the cross-correlation function for leak detection in a plastic water distribution pipe using acoustic signals“. In: *Applied Acoustics* 127 (2017), S. 24–33.

[GBC16] Ian Goodfellow, Yoshua Bengio und Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[GCG21] Yuan Gong, Yu-An Chung und James Glass. „AST: Audio Spectrogram Transformer“. In: *Proc. Interspeech 2021*. 2021, S. 571–575. DOI: 10.21437/Interspeech.2021-698.

[GDG03] Robert Givan, Thomas Dean und Matthew Greig. „Equivalence notions and model minimization in Markov decision processes“. In: *Artificial Intelligence* 147.1-2 (2003), S. 163–223. DOI: 10.1016/S0004-3702(02)00376-4.

[Gel+19] Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum und Marc G. Bellemare. „DeepMDP: Learning Continuous Latent Space Models for Representation Learning“. In: *Proceedings of the 36th International Conference on Machine Learning*. Bd. 97. Proceedings of Machine Learning Research. PMLR, Sep. 2019, S. 2170–2179.

[Gem+17] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal und Marvin Ritter. „Audio Set: An ontology and human-labeled dataset for audio events“. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, S. 776–780. DOI: 10.1109/ICASSP.2017.7952261.

[Gir+20] Ritwik Giri, Fangzhou Cheng, Karim Helwani, Srikanth V Tenneti, Umut Isik und Arvindh Krishnaswamy. „Group masked autoencoder based density estimator for audio anomaly detection“. In: *Detection and Classification of Acoustic Scenes and Events Workshop*. 2020.

[GJM13] Alex Graves, Navdeep Jaitly und Abdel-rahman Mohamed. „Hybrid speech recognition with Deep Bidirectional LSTM“. In: *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. 2013, S. 273–278. DOI: 10.1109/ASRU.2013.6707742.

[GKH19] Ritwik Giri, Arvindh Krishnaswamy und Karim Helwani. „Robust non-negative block sparse coding for acoustic novelty detection“. In: *Detection and Classification of Acoustic Scenes and Events Workshop*. 2019.

[GL16] Aditya Grover und Jure Leskovec. „Node2Vec: Scalable Feature Learning for Networks“. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. New York, NY, USA: ACM, 2016, S. 855–864. DOI: 10.1145/2939672.2939754.

[Gon+22] Yuan Gong, Cheng-I Lai, Yu-An Chung und James Glass. „Ssast: Self-supervised audio spectrogram transformer“. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Bd. 36. 10. 2022, S. 10699–10709. DOI: 10.1609/aaai.v36i10.21315.

[GP07] Todor Ganchev und Ilyas Potamitis. „Automatic acoustic identification of singing insects“. In: *Bioacoustics* 16.3 (2007), S. 281–328. DOI: 10.1080/09524622.2007.9753582.

[Gri+20] Jean-Bastien Grill u. a. „Bootstrap Your Own Latent a New Approach to Self-Supervised Learning“. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[Gup+11] Mangesh Gupte, Pravin Shankar, Jing Li, S. Muthukrishnan und Liviu Iftode. „Finding Hierarchy in Directed Online Social Networks“. In: *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. New York, NY, USA: ACM, 2011, S. 557–566. DOI: 10.1145/1963405.1963484.

[GW08] Andreas Griewank und Andrea Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM, 2008.

[HA10] Lars Magnus Hvattum und Halvard Arntzen. „Using ELO ratings for match result prediction in association football“. In: *International Journal of Forecasting* 26 (Juli 2010), S. 460–470. DOI: 10.1016/j.ijforecast.2009.10.002.

[Hai+21] Tom Haider, Felippe Schmoeller Roza, Dirk Eilers, Karsten Roscher und Stephan Günnemann. „Domain Shifts in Reinforcement Learning: Identifying Disturbances in Environments“. In: *AISafety@IJCAI*. 2021.

[Han+20] Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A Efros, Lerrel Pinto und Xiaolong Wang. „Self-Supervised Policy Adaptation during Deployment“. In: *International Conference on Learning Representations (ICLR)*. 2020.

[Hay+18] Tomoki Hayashi, Tatsuya Komatsu, Reishi Kondo, Tomoki Toda und Kazuya Takeda. „Anomalous sound event detection based on wavenet“. In: *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE. 2018, S. 2494–2498. DOI: 10.23919/EUSIPCO.2018.8553423.

[HB17] Xun Huang und Serge Belongie. „Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization“. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, S. 1510–1519. DOI: 10.1109/ICCV.2017.167.

[HBZ04] Eric A. Hansen, Daniel S. Bernstein und Shlomo Zilberstein. „Dynamic Programming for Partially Observable Stochastic Games“. In: *Proceedings of the 19th National Conference on Artifical Intelligence*. AAAI'04. AAAI Press, 2004, S. 709–715.

[He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren und Jian Sun. „Deep Residual Learning for Image Recognition“. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, S. 770–778. DOI: 10.1109/CVPR.2016.90.

[He+20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie und Ross Girshick. „Momentum Contrast for Unsupervised Visual Representation Learning“. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, S. 9726–9735. DOI: 10.1109/CVPR42600.2020.00975.

[He+22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár und Ross Girshick. „Masked Autoencoders Are Scalable Vision Learners“. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, S. 15979–15988. DOI: 10.1109/CVPR52688.2022.01553.

[Heb49] Donald O Hebb. „The first stage of perception: growth of the assembly“. In: *The Organization of Behavior* 4 (1949), S. 60–78. DOI: 10.7551/mitpress/4943.003.0006.

[Hei+15] Stefanie Heinicke, Ammie K Kalan, Oliver JJ Wagner, Roger Mundry, Hanna Lukashevich und Hjalmar S Kühl. „Assessing the performance of a semi-automated acoustic monitoring system for primates“. In: *Methods in Ecology and Evolution* 6.7 (2015), S. 753–763. DOI: 10.1111/2041-210X.12384.

[Her+17a] Shawn Hershey u. a. „CNN Architectures for Large-Scale Audio Classification“. In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017. DOI: 10.1109/ICASSP.2017.7952132.

[Her+17b] Shawn Hershey u. a. „CNN architectures for large-scale audio classification“. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, S. 131–135. DOI: 10.1109/ICASSP.2017.7952132.

[Hoc91] Sepp Hochreiter. „Untersuchungen zu dynamischen neuronalen Netzen“. In: *Diploma, Technische Universität München* 1 (1991).

[Hor+12] Kurt Hornik, Ingo Feinerer, Martin Kober und Christian Buchta. „Spherical k-means clustering“. In: *Journal of statistical software* 50 (2012), S. 1–22. DOI: 10.18637/jss.v050.i10.

[How+17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto und Hartwig Adam. „Mobilenets: Efficient convolutional neural networks for mobile vision applications“. In: *arXiv preprint arXiv:1704.04861* (2017).

[HS97] Sepp Hochreiter und Jürgen Schmidhuber. „Long short-term memory“. In: *Neural computation* 9.8 (1997), S. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[HSW89] Kurt Hornik, Maxwell Stinchcombe und Halbert White. „Multilayer feedforward networks are universal approximators“. In: *Neural networks* 2.5 (1989), S. 359–366. DOI: 10.1016/0893-6080(89)90020-8.

[Hua+17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten und Kilian Q. Weinberger. „Densely Connected Convolutional Networks“. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, S. 2261–2269. DOI: 10.1109/CVPR.2017.243.

[Ian+16] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally und Kurt Keutzer. „Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size, 2016“. In: *arXiv preprint arXiv:1602.07360* 10 (2016).

[Ill+20] Steffen Illium, Robert Müller, Andreas Sedlmeier und Claudia Linnhoff-Popien. „Surgical Mask Detection with Convolutional Neural Networks and Data Augmentations on Spectrograms“. In: *Proc. Interspeech 2020*. 2020, S. 2052–2056. DOI: 10.21437/Interspeech.2020-1692.

[IS15] Sergey Ioffe und Christian Szegedy. „Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift“. In: *Proceedings of the 32nd International Conference on Machine Learning*. Bd. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, Juli 2015, S. 448–456.

[Jaq+19] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, Dj Strouse, Joel Z. Leibo und Nando De Freitas. „Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning“. In: *Proceedings of the 36th International Conference on Machine Learning*. Hrsg. von Kamalika Chaudhuri und Ruslan Salakhutdinov. Bd. 97. Proceedings of Machine Learning Research. PMLR, Sep. 2019, S. 3040–3049.

[JD88] Anil K. Jain und Richard C. Dubes. *Algorithms for Clustering Data*. USA: Prentice-Hall, Inc., 1988.

[Jum+21] John Jumper u. a. „Highly accurate protein structure prediction with AlphaFold“. In: *Nature* 596.7873 (2021), S. 583–589. DOI: 10.1038/s41586-021-03819-2.

[Kan+17] Jiheon Kang, Youn-Jong Park, Jaeho Lee, Soo-Hyun Wang und Doo-Seop Eom. „Novel leakage detection by ensemble CNN-SVM and graph-based localization in water distribution systems“. In: *IEEE Transactions on Industrial Electronics* 65.5 (2017), S. 4279–4289. DOI: 10.1109/TIE.2017.2764861.

[Kaw+19] Y. Kawaguchi, R. Tanabe, T. Endo, K. Ichige und K. Hamada. „Anomaly Detection Based on an Ensemble of Dereverberation and Anomalous Sound Extraction“. In: *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, S. 865–869.

[KB15] Diederik P Kingma und Jimmy Ba. „Adam: A Method for Stochastic Optimization“. In: *International Conference on Learning Representations (ICLR), ICLR 2015*. 2015.

[Khe+20] Khimya Khetarpal, Matthew Riemer, Irina Rish und Doina Precup. „Towards Continual Reinforcement Learning: A Review and Perspectives“. In: *J. Artif. Intell. Res.* 75 (2020), S. 1401–1476. DOI: 10.1613/jair.1.13673.

[KL51] Solomon Kullback und Richard A Leibler. „On information and sufficiency“. In: *The annals of mathematical statistics* 22.1 (1951), S. 79–86. DOI: 10.1214/aoms/1177729694.

[KLA21] T. Karras, S. Laine und T. Aila. „A Style-Based Generator Architecture for Generative Adversarial Networks“. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 43.12 (Dez. 2021). DOI: 10.1109/TPAMI.2020.2970919.

[Koi+17] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu und Noboru Harada. „Optimizing acoustic feature extractor for anomalous sound detection based on Neyman-Pearson lemma“. In: *2017 25th European Signal Processing Conference (EUSIPCO)*. IEEE. 2017, S. 698–702. DOI: 10.23919/EUSIPCO.2017.8081297.

[Koi+19] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada und Keisuke Imoto. „ToyADMOS: A dataset of miniature-machine operating sounds for anomalous sound detection“. In: *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE. 2019, S. 313–317. DOI: 10.1109/WASPAA.2019.8937164.

[Koi+20a] Yuma Koizumi, Masahiro Yasuda, Shin Murata, Shoichiro Saito, Hisashi Uematsu und Noboru Harada. „SPIDERnet: Attention Network For One-Shot Anomaly Detection In Sounds“. In: *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2020), S. 281–285. DOI: 10.1109/ICASSP40776.2020.9053620.

[Koi+20b] Yuma Koizumi u. a. „Description and Discussion on DCASE2020 Challenge Task2: Unsupervised Anomalous Sound Detection for Machine Condition Monitoring“. In: *arXiv preprint arXiv:2006.05822* (2020).

[Kon+20] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang und Mark D Plumbley. „Panns: Large-scale pretrained audio neural networks for audio pattern recognition“. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28 (2020), S. 2880–2894. DOI: 10.1109/TASLP.2020.3030497.

[KR08] Leonard Kaufman und Peter J. Rousseeuw. „Partitioning Around Medoids (Program PAM)“. In: *Finding Groups in Data*. John Wiley & Sons, Inc., 2008, S. 68–125. DOI: 10.1002/9780470316801.ch2.

[KSH12] Alex Krizhevsky, Ilya Sutskever und Geoffrey E Hinton. „ImageNet Classification with Deep Convolutional Neural Networks“. In: *Advances in Neural Information Processing Systems*. Bd. 25. Curran Associates, Inc., 2012.

[KSL19] Simon Kornblith, Jonathon Shlens und Quoc V. Le. „Do Better ImageNet Models Transfer Better?“ In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, S. 2656–2666. DOI: 10.1109/CVPR.2019.00277.

[KW14] Diederik P. Kingma und Max Welling. „Auto-Encoding Variational Bayes“. In: *International Conference on Learning Representations (ICLR)*. 2014.

[Lan+22] Lauro Langosco Di Langosco, Jack Koch, Lee D Sharkey, Jacob Pfau und David Krueger. „Goal Misgeneralization in Deep Reinforcement Learning“. In: *Proceedings of the 39th International Conference on Machine Learning*. Bd. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, S. 12004–12019.

[Law+09] Edith Law, Kris West, Michael I. Mandel, Mert Bay und J. Stephen Downie. „Evaluation of Algorithms Using Games: The Case of Music Tagging“. In: *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe International Conference Center, Kobe, Japan, October 26-30, 2009*. International Society for Music Information Retrieval, 2009, S. 387–392.

[LeC+99] Yann LeCun, Patrick Haffner, Léon Bottou und Yoshua Bengio. „Object recognition with gradient-based learning“. In: *Shape, contour and grouping in computer vision*. Springer, 1999, S. 319–345. DOI: 10.1007/3-540-46805-6_19.

[Lee+20] Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee und Jinwoo Shin. „Context-aware Dynamics Model for Generalization in Model-Based Reinforcement Learning“. In: *Proceedings of the 37th International Conference on Machine Learning*. Hrsg. von

Hal Daumé III und Aarti Singh. Bd. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, S. 5757–5766.

[LH18] Ilya Loshchilov und Frank Hutter. „Decoupled Weight Decay Regularization“. In: *International Conference on Learning Representations (ICLR)*. 2018.

[Lin+17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He und Piotr Dollár. „Focal Loss for Dense Object Detection“. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, S. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[Lin+21] Toru Lin, Jacob Huh, Christopher Stauffer, Ser Nam Lim und Phillip Isola. „Learning to Ground Multi-Agent Communication with Autoencoders“. In: *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021), S. 15230–15242.

[Lit94] Michael L Littman. „Markov games as a framework for multi-agent reinforcement learning“. In: *Machine learning proceedings 1994*. Elsevier, 1994, S. 157–163.

[Liu+19] Yun Liu u. a. „Artificial intelligence–based breast cancer nodal metastasis detection: Insights into the black box for pathologists“. In: *Archives of pathology & laboratory medicine* 143.7 (2019), S. 859–868. DOI: 10.5858/arpa.2018-0147-OA.

[Liu+22] Shuo Liu, Adria Mallol-Ragolta, Emilia Parada-Cabaleiro, Kun Qian, Xin Jing, Alexander Kathan, Bin Hu und Björn W. Schuller. „Audio self-supervised learning: A survey“. In: *Patterns* 3.12 (2022), S. 100616. DOI: 10.1016/j.patter.2022.100616.

[LKA17] Jake Lever, Martin Krzywinski und Naomi Altman. „Points of significance: Principal component analysis“. In: *Nature methods* 14.7 (2017), S. 641–643. DOI: 10.1038/nmeth.4346.

[Llo82] S. Lloyd. „Least squares quantization in PCM“. In: *IEEE Transactions on Information Theory* 28.2 (1982), S. 129–137. DOI: 10.1109/TIT.1982.1056489.

[Loc+20] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy und Thomas Kipf. „Object-Centric Learning with Slot Attention“. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[Low+17] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel und Igor Mordatch. „Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments“. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, S. 6382–6393.

[LPY17] Hoang M Le, Carr Peter und Yisong Yue. „Data-Driven Ghosting using Deep Imitation Learning“. In: *MIT Sloan Sports Analytics Conference* (2017), S. 1–15.

[LTZ08] Fei Tony Liu, Kai Ming Ting und Zhi-Hua Zhou. „Isolation forest“. In: *2008 Eighth IEEE International Conference on Data Mining*. IEEE. 2008, S. 413–422. DOI: 10.1109/ICDM.2008.17.

[LZH10] Christoph Leitner, Achim Zeileis und Kurt Hornik. „Forecasting sports tournaments by ratings of (prob) abilities: A comparison for the EURO 2008“. In: *International Journal of Forecasting* 26.3 (2010), S. 471–481. DOI: 10.1016/j.ijforecast.2009.10.001.

[MA18] Igor Mordatch und Pieter Abbeel. „Emergence of Grounded Compositional Language in Multi-Agent Populations“. In: AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018.

[Mak+15] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow und Brendan Frey. „Adversarial Autoencoders“. In: (2015).

[Man+19] Daniel J Mankowitz u. a. „Robust Reinforcement Learning for Continuous Control with Model Misspecification“. In: *International Conference on Learning Representations (ICLR)*. 2019.

[Man+21] Khushdeep Singh Mann, Steffen Schneider, Alberto Chiappa, Jin Hwa Lee, Matthias Bethge, Alexander Mathis und Mackenzie W Mathis. „Out-of-distribution generalization of internal models is correlated with reward“. In: *Self-Supervision for Reinforcement Learning Workshop-ICLR 2021*. 2021.

[Mar+15] Erik Marchi, Fabio Vesperini, Florian Eyben, Stefano Squartini und Björn Schuller. „A Novel Approach for Automatic Acoustic Novelty Detection Using a Denoising Autoencoder with Bidirectional LSTM Neural Networks“. In: *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2015, S. 1996–2000. DOI: 10.1109/ICASSP.2015.7178320.

[MB04] JM Muggleton und MJ Brennan. „Leak noise propagation and attenuation in submerged plastic water pipes“. In: *Journal of Sound and Vibration* 278.3 (2004), S. 527–537. DOI: 10.1016/j.jsv.2003.10.052.

[MH08] Laurens van der Maaten und Geoffrey Hinton. „Visualizing Data using t-SNE“. In: *Journal of Machine Learning Research* 9.86 (2008), S. 2579–2605.

[Mik+13a] Tomas Mikolov, Kai Chen, Greg Corrado und Jeffrey Dean. „Efficient Estimation of Word Representations in Vector Space“. In: *1st International Conference on Learning Representations (ICLR)*. 2013.

[Mik+13b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado und Jeff Dean. „Distributed Representations of Words and Phrases and their Compositionality“. In: *Advances in Neural Information Processing Systems (NeurIPS) 26*. Curran Associates, Inc., 2013, S. 3111–3119.

[Mik+13c] Tomás Mikolov, Kai Chen, Greg Corrado und Jeffrey Dean. „Efficient Estimation of Word Representations in Vector Space“. In: *International Conference on Learning Representations (ICLR)*. 2013.

[MIL21a] Robert Müller, Steffen Illium und Claudia Linnhoff-Popien. „A Deep and Recurrent Architecture for Primate Vocalization Classification“. In: *Proc. Interspeech 2021*. 2021, S. 461–465. DOI: 10.21437/Interspeech.2021-1274.

[MIL21b] Robert Müller, Steffen Illium und Claudia Linnhoff-Popien. „Deep Recurrent Interpolation Networks for Anomalous Sound Detection“. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. 2021, S. 1–7. DOI: 10.1109/IJCNN52387.2021.9533560.

[Min] T Ming. „Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents“. In: *Proceedings of the Tenth International Conference on Machine Learning*, S. 330–337.

[MK19] Maarten Meire und Peter Karsmakers. „Comparison of Deep Autoencoder Architectures for Real-time Acoustic Based Anomaly Detection in Assets“. In: *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. Bd. 2. IEEE. 2019, S. 786–790. DOI: 10.1109/IDAACS.2019.8924301.

[MNG17] Lars Mescheder, Sebastian Nowozin und Andreas Geiger. „Adversarial Variational Bayes: Unifying Variational Autoencoders and Generative Adversarial Networks“. In: *Proceedings of the 34th International Conference on Machine Learning*. ICML'17. JMLR, 2017, S. 2391–2400.

[Moo+65] Gordon E Moore u. a. *Cramming more components onto integrated circuits*. 1965. DOI: 10.1109/JPROC.1998.658762.

[MPH22] Jitendra Singh Malik, Guansong Pang und Anton van den Hengel. „Deep Learning for Hate Speech Detection: A Comparative Study“. In: *arXiv preprint arXiv:2202.09517* (2022).

[Mül+20] Robert Müller, Stefan Langer, Fabian Ritz, Christoph Roch, Steffen Illium und Claudia Linnhoff-Popien. „Soccer Team Vectors“. In: *Machine Learning and Knowledge Discovery in Databases*. Cham: Springer International Publishing, 2020, S. 247–257. DOI: 10.1007/978-3-030-43887-6_19.

[Mül+21a] R. Müller, Steffen Illium., Fabian Ritz. und Kyrill Schmid. „Analysis of Feature Representations for Anomalous Sound Detection“. In: *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART,* INSTICC. SciTePress, 2021, S. 97–106. DOI: 10.5220/0010226800970106.

[Mül+21b] R. Müller, Steffen Illium., Fabian Ritz., Tobias Schröder., Christian Platschek., Jörg Ochs. und Claudia Linnhoff-Popien. „Acoustic Leak Detection in Water Networks“. In: *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART,* INSTICC. SciTePress, 2021, S. 306–313. DOI: 10.5220/0010295403060313.

[Mül+21c] R. Müller, Fabian Ritz., Steffen Illium. und Claudia Linnhoff-Popien. „Acoustic Anomaly Detection for Machine Sounds based on Image Transfer Learning“. In: *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART,* INSTICC. SciTePress, 2021, S. 49–56. DOI: 10.5220/0010185800490056.

[Mül+21d] Robert Müller., Steffen Illium., Fabian Ritz. und Kyrill Schmid. „Analysis of Feature Representations for Anomalous Sound Detection“. In: *Proceedings of the 13th International Conference on Agents and Artificial Intelligence*. INSTICC. SciTePress, 2021, S. 97–106. DOI: 10.5220/0010226800970106.

[Mül+22] Robert Müller, Steffen Illium, Thomy Phan, Tom Haider und Claudia Linnhoff-Popien. „Towards Anomaly Detection in Reinforcement Learning“. In: *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*. AAMAS '22. Richland, SC: International Foundation for Autonomous Agents und Multiagent Systems, 2022, S. 1799–1803.

[MZ13] Alexander Mielke und Klaus Zuberbühler. „A method for automated individual, species and call type recognition in free-ranging animals“. In: *Animal Behaviour* 86.2 (2013), S. 475–482. DOI: 10.1016/j.anbehav.2013.04.017.

[Nab+21] Zaid Nabulsi u. a. „Deep learning for distinguishing normal versus abnormal chest radiographs and generalization to two unseen diseases tuberculosis and COVID-19“. In: *Scientific reports* 11.1 (2021), S. 1–15. DOI: 10.1038/s41598-021-93967-2.

[Nag+21] Arsha Nagrani, Shan Yang, Anurag Arnab, Aren Jansen, Cordelia Schmid und Chen Sun. „Attention Bottlenecks for Multimodal Fusion“. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Bd. 34. Curran Associates, Inc., 2021, S. 14200–14213.

[Nak+16] Yasutaka Nakajima, Taisuke Naito, Norihito Sunago, Toshiya Ohshima und Nobutaka Ono. „DNN-based environmental sound recognition with real-recorded and artificially-mixed training data“. In: *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*. Bd. 253. Institute of Noise Control Engineering. 2016, S. 1832–1841.

[NH10] Vinod Nair und Geoffrey E. Hinton. „Rectified Linear Units Improve Restricted Boltzmann Machines“. In: *Proceedings of the 27th international conference on machine learning*. 2010, S. 807–814.

[NI02] Qing-Qing Ni und Masaharu Iwamoto. „Wavelet transform of acoustic emission signals in failure of model composites“. In: *Engineering Fracture Mechanics* 69.6 (2002), S. 717–728. DOI: 10.1016/S0013-7944(01)00105-9.

[NRB18] Stefan Neumann, Julian Ritter und Kailash Budhathoki. „Ranking the teams in european football leagues with agony“. In: *International Workshop on Machine Learning and Data Mining for Sports Analytics*. Springer. 2018, S. 55–66. DOI: 10.1007/978-3-030-17274-9.

[NS19] Toan Q. Nguyen und Julian Salazar. „Transformers without Tears: Improving the Normalization of Self-Attention“. In: *Proceedings of the 16th International Conference on Spoken Language Translation*. Hong Kong: Association for Computational Linguistics, 2019.

[Ola+20] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov und Shan Carter. „An Overview of Early Vision in InceptionV1“. In: *Distill* 5.4 (2020). DOI: 10.23915/distill.00024.002.

[OMS17] Chris Olah, Alexander Mordvintsev und Ludwig Schubert. „Feature visualization“. In: *Distill* 2.11 (2017). DOI: 10.23915/distill.00007.

[Oor+] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior und Koray Kavukcuoglu. „WaveNet: A Generative Model for Raw Audio“. In: *9th ISCA Speech Synthesis Workshop*, S. 125–125.

[Pan+21] Guansong Pang, Chunhua Shen, Longbing Cao und Anton Van Den Hengel. „Deep Learning for Anomaly Detection: A Review“. In: *ACM Comput. Surv.* 54.2 (März 2021). DOI: 10.1145/3439950.

[Pan+22] M. Pandy, A. Agostinelli, J. Uijlings, V. Ferrari und T. Mensink. „Transferability Estimation using Bhattacharyya Class Separability“. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Juni 2022, S. 9162–9172. DOI: 10.1109/CVPR52688.2022.00896.

[Par+19] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk und Quoc V. Le. „SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition“. In: *Proc. Interspeech 2019*. 2019, S. 2613–2617. DOI: 10.21437/Interspeech.2019-2680.

[Pas+19] Adam Paszke u. a. „PyTorch: An Imperative Style, High-Performance Deep Learning Library“. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[PAS14] Bryan Perozzi, Rami Al-Rfou und Steven Skiena. „DeepWalk: Online Learning of Social Representations“. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. New York, NY, USA: Association for Computing Machinery, 2014, S. 701–710. DOI: 10.1145/2623330.2623732.

[Pat+17] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros und Trevor Darrell. „Curiosity-Driven Exploration by Self-Supervised Prediction“. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. Proceedings of Machine Learning Research. PMLR, 2017, S. 2778–2787.

[Ped+11] F. Pedregosa u. a. „Scikit-learn: Machine Learning in Python“. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830.

[Pel18] Konstantinos Pelechrinis. „LinNet: Probabilistic Lineup Evaluation Through Network Embedding“. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2018, S. 20–36. DOI: 10.1007/978-3-030-10997-4.

[Pet+18a] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee und Luke Zettlemoyer. „Deep Contextualized Word Representations“. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*. 2018, S. 2227–2237. DOI: 10.18653/v1/n18-1202.

[Pet+18b] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee und Luke Zettlemoyer. „Deep Contextualized Word Representations“. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, Volume 1*. Association for Computational Linguistics, 2018, S. 2227–2237. DOI: 10.18653/v1/n18-1202.

[Pha+17] Huy Phan, Philipp Koch, Fabrice Katzberg, Marco Maass, Radoslaw Mazur und Alfred Mertins. „Audio Scene Classification with Deep Recurrent Neural Networks“. In: *Proc. Interspeech 2017*. 2017, S. 3043–3047. DOI: 10.21437/Interspeech.2017-101.

[Pha+21] Thomy Phan, Lenz Belzner, Thomas Gabor, Andreas Sedlmeier, Fabian Ritz und Claudia Linnhoff-Popien. „Resilient Multi-Agent Reinforcement Learning with Adversarial Value Decomposition“. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Bd. 35. 13. 2021, S. 11308–11316. DOI: 10.1609/aaai.v35i13.17348.

[Pin+17] Lerrel Pinto, James Davidson, Rahul Sukthankar und Abhinav Gupta. „Robust Adversarial Reinforcement Learning“. In: *Proceedings of the 34th International Conference on Machine Learning*. Bd. 70. Proceedings of Machine Learning Research. PMLR, Juni 2017, S. 2817–2826.

[PMC18] Nirosha Priyadarshani, Stephen Marsland und Isabel Castro. „Automated birdsong recognition in complex acoustic environments: a review“. In: *Journal of Avian Biology* 49.5 (2018), jav–01447. DOI: 10.1111/jav.01447.

[PNH20] Hyunjong Park, Jongyoun Noh und Bumsub Ham. „Learning Memory-Guided Normality for Anomaly Detection“. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, S. 14360–14369. DOI: 10.1109/CVPR42600.2020.01438.

[PS19] Jordi Pons und Xavier Serra. „musicnn: Pre-trained convolutional neural networks for music audio tagging“. In: *arXiv preprint arXiv:1909.06654* (2019).

[PS20] Dishant Parikh und Saurabh Sachdev. „Improving the efficiency of spectral features extraction by structuring the audio files“. In: *2020 IEEE-HYDCON*. IEEE. 2020, S. 1–5.

[Pur+01] William K Purves, David E Sadava, Gordon H Orians und H Craig Heller. *Life: the science of biology*. Macmillan, 2001. DOI: 10.1002/cbf.1179.

[Pur+19] Harsh Purohit, Ryo Tanabe, Kenji Ichige, Takashi Endo, Yuki Nikaido, Kaori Suefusa und Yohei Kawaguchi. „MIMII Dataset: Sound Dataset for Malfunctioning Indsutrial Machine Investigation and Inspection“. In: *Acoustic Scenes and Events 2019 Workshop (DCASE2019)*. 2019, S. 209. DOI: 10.5281/zenodo.3384388.

[PY09] Sinno Jialin Pan und Qiang Yang. „A survey on transfer learning“. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), S. 1345–1359. DOI: 10.1109/TKDE.2009.191.

[Qia+16] Yanmin Qian, Mengxiao Bi, Tian Tan und Kai Yu. „Very deep convolutional neural networks for noise robust speech recognition“. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.12 (2016), S. 2263–2276. DOI: 10.1109/TASLP.2016.2602884.

[RCC17] Rajeev Ranjan, Carlos D Castillo und Rama Chellappa. „L2-constrained softmax loss for discriminative face verification“. In: *arXiv preprint arXiv:1703.09507* (2017).

[Ren+18] Zhao Ren, Nicholas Cummins, Vedhas Pandit, Jing Han, Kun Qian und Björn Schuller. „Learning image-based representations for heart sound classification“. In: *Proceedings of the 2018 international conference on digital health*. 2018, S. 143–147. DOI: 10.1145/3194658.3194671.

[RH21] Tal Reiss und Yedid Hoshen. „Mean-Shifted Contrastive Loss for Anomaly Detection“. In: *arXiv preprint arXiv:2106.03844* (2021).

[RHW86] David E Rumelhart, Geoffrey E Hinton und Ronald J Williams. „Learning representations by back-propagating errors“. In: *Nature* 323.6088 (1986), S. 533–536. DOI: /10.1038/323533a0.

[RM19] Ellen Rushe und Brian Mac Namee. „Anomaly Detection in Raw Audio Using Deep Autoregressive Networks“. In: *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, S. 3597–3601. DOI: 10.1109/ICASSP.2019.8683414.

[Ros58] Frank Rosenblatt. „The perceptron: a probabilistic model for information storage and organization in the brain.“ In: *Psychological review* 65.6 (1958), S. 386. DOI: 10.1037/h0042519.

[Ruf+18] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller und Marius Kloft. „Deep One-Class Classification“. In: *Proceedings of the 35th International Conference on Machine Learning*. Bd. 80. Proceedings of Machine Learning Research. PMLR, Okt. 2018, S. 4393–4402.

[Ruf+19] Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake und Marius Kloft. „Self-Attentive, Multi-Context One-Class Classification for Unsupervised Anomaly Detection on Text“. In: *Proceedings of the 57th Conference of the Association for Computational Linguistics*. Juli 2019, S. 4061–4071. DOI: 10.18653/v1/P19-1398.

[RV16] Danilo Russo und Christian C Voigt. „The use of automated identification of bat echolocation calls in acoustic monitoring: A cautionary note for a sound analysis“. In: *Ecological Indicators* 66 (2016), S. 598–602. DOI: 10.1016/j.ecolind.2016.02.036.

[S H54] Zellig S. Harris. „Distributional Structure“. In: *Word* 10 (Aug. 1954), S. 146–162. DOI: 10.1007/978-94-009-8467-7_1.

[Sam59] A. L. Samuel. „Some Studies in Machine Learning Using the Game of Checkers“. In: *IBM Journal of Research and Development* 3.3 (1959), S. 210–229. DOI: 10.1147/rd.33.0210.

[SB17] Justin Salamon und Juan Pablo Bello. „Deep convolutional neural networks and data augmentation for environmental sound classification“. In: *IEEE Signal Processing Letters* 24.3 (2017), S. 279–283. DOI: 10.1109/LSP.2017.2657381.

[Sch+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford und Oleg Klimov. „Proximal policy optimization algorithms“. In: *arXiv preprint arXiv:1707.06347* (2017).

[Sch+21] Björn W. Schuller u. a. „The INTERSPEECH 2021 Computational Paralinguistics Challenge: COVID-19 Cough, COVID-19 Speech, Escalation & Primates“. In: *Proc. Interspeech 2021*. 2021, S. 431–435. DOI: 10.21437/Interspeech.2021-19.

[Sch+99] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor und John Platt. „Support Vector Method for Novelty Detection“. In: *Proceedings of the 12th International Conference on Neural Information Processing Systems*. NIPS'99. Cambridge, MA, USA: MIT Press, 1999, S. 582–588.

[SCS18] W. Sultani, C. Chen und M. Shah. „Real-World Anomaly Detection in Surveillance Videos“. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Juni 2018, S. 6479–6488. DOI: 10.1109/CVPR.2018.00678.

[Sed+20a] Andreas Sedlmeier, Thomas Gabor, Thomy Phan und Lenz Belzner. „Uncertainty-Based Out-of-Distribution Detection in Deep Reinforcement Learning“. In: *Digitale Welt* 4.1 (2020), S. 74–78. DOI: 10.1007/s42354-019-0238-z.

[Sed+20b] Andreas Sedlmeier, Robert Müller, Steffen Illium und Claudia Linnhoff-Popien. „Policy Entropy for Out-of-Distribution Classification“. In: *International Conference on Artificial Neural Networks*. Springer. 2020, S. 420–431. DOI: 10.1007/978-3-030-61616-8_34.

[Seo+21] Younggyo Seo, Lili Chen, Jinwoo Shin, Honglak Lee, Pieter Abbeel und Kimin Lee. „State Entropy Maximization with Random Encoders for Efficient Exploration“. In: *Proceedings of the 38th International Conference on Machine Learning*. Bd. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, S. 9443–9454.

[Sha48] Claude Elwood Shannon. „A mathematical theory of communication“. In: *The Bell system technical journal* 27.3 (1948), S. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[Sha49] Claude E Shannon. „Communication in the Presence of Noise“. In: *Proceedings of the IRE* 37.1 (1949), S. 10–21. DOI: 10.1109/JRPROC.1949.232969.

[Sha53] Lloyd S Shapley. „Stochastic games“. In: *Proceedings of the national academy of sciences* 39.10 (1953), S. 1095–1100. DOI: 10.1073/pnas.39.10.1095.

[Sil+16] David Silver u. a. „Mastering the game of Go with deep neural networks and tree search“. In: *Nature* 529.7587 (2016), S. 484–489. DOI: 10.1038/nature16961.

[Sil+17] David Silver u. a. „Mastering the game of go without human knowledge“. In: *Nature* 550.7676 (2017), S. 354–359. DOI: 0.1038/nature24270.

[Soc+15] Joan Claudi Socoró, Gerard Ribera, Xavier Sevillano und Francesc Alías. „Development of an Anomalous Noise Event Detection Algorithm for dynamic road traffic noise mapping“. In: *Proceedings of the 22nd International Congress on Sound and Vibration (ICSV22), Florence, Italy*. 2015, S. 12–16.

[Sod+22] Shagun Sodhani, Franziska Meier, Joelle Pineau und Amy Zhang. „Block Contextual MDPs for Continual Learning“. In: *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*. Bd. 168. Proceedings of Machine Learning Research. PMLR, 23–24 Jun 2022, S. 608–623.

[Sou+21] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak und Even Oldridge. „Transformers4Rec: Bridging the Gap between NLP and Sequential/Session-Based Recommendation“. In: *Fifteenth ACM Conference on Recommender Systems*. 2021, S. 143–153. DOI: 10.1145/3460231.3474255.

[SS17] Maximilian Schmitt und Björn Schuller. „openXBOW – Introducing the Passau Open-Source Crossmodal Bag-of-Words Toolkit“. In: *Journal of Machine Learning Research* 18.96 (2017), S. 1–5.

[SS19] Maximilian Schmitt und Björn Schuller. „End-to-end Audio Classification with Small Datasets – Making It Work“. In: *2019 27th European Signal Processing Conference (EUSIPCO)*. 2019, S. 1–5. DOI: 10.23919/EUSIPCO.2019.8902712.

[SS20] Oren Salzman und Roni Stern. „Research Challenges and Opportunities in Multi-Agent Path Finding and Multi-Agent Pickup and Delivery Problems“. In: AAMAS '20. Richland, SC: International Foundation for Autonomous Agents und Multiagent Systems, 2020, S. 1711–1715.

[SSF16] Sainbayar Sukhbaatar, Arthur Szlam und Rob Fergus. „Learning Multiagent Communication with Backpropagation“. In: NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, S. 2252–2260.

[SSS16] Shai Shalev-Shwartz, Shaked Shammah und Amnon Shashua. „Safe, Multi-Agent, Reinforcement Learning for Autonomous Driving“. In: *arXiv preprint arXiv:1610.03295* (2016).

[Sto+07] I. Stoianov, L. Nachman, S. Madden, T. Tokmouline und M. Csail. „PIPENET: A Wireless Sensor Network for Pipeline Monitoring“. In: *2007 6th International Symposium on Information Processing in Sensor Networks*. Apr. 2007, S. 264–273. DOI: 10.1109/IPSN.2007.4379686.

[Sto+15] Dan Stowell, Dimitrios Giannoulis, Emmanouil Benetos, Mathieu Lagrange und Mark D Plumbley. „Detection and classification of acoustic scenes and events“. In: *IEEE Transactions on Multimedia* 17.10 (2015), S. 1733–1746. DOI: 10.1109/TASLP.2017.2778423.

[Sue+20] Kaori Suefusa, Tomoya Nishida, Harsh Purohit, Ryo Tanabe, Takashi Endo und Yohei Kawaguchi. „Anomalous Sound Detection Based on Interpolation Deep Neural Network“. In: *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, S. 271–275. DOI: 10.1109/ICASSP40776.2020.9054344.

[SVN37] Stanley Smith Stevens, John Volkmann und Edwin Broomell Newman. „A scale for the measurement of the psychological magnitude pitch“. In: *The journal of the acoustical society of america* 8.3 (1937), S. 185–190. DOI: 10.1121/1.1915893.

[SWM18] Pressedienst SWM. *Die SWM vernetzen München: LoRa-Netz am Start für das Internet der Dinge*. https://www.swm.de/dam/swm/pressemitteilungen/2018/06/20180604-swm-bauen-lora-netz-auf.pdf. Juni 2018.

[SZ15] Karen Simonyan und Andrew Zisserman. „Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations (ICLR), ICLR 2015*. 2015.

[Tac+20] Jihoon Tack, Sangwoo Mo, Jongheon Jeong und Jinwoo Shin. „CSI: Novelty Detection via Contrastive Learning on Distributionally Shifted Instances". In: *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), S. 11839–11852.

[Tal05] Maggie Ed Tallerman. *Language origins: Perspectives on evolution*. Oxford University Press, 2005.

[Tin+19] LL Ting, JY Tey, AC Tan, YJ King und AR Faidz. „Improvement of acoustic water leak detection based on dual tree complex wavelet transform-correlation method". In: *IOP Conference Series: Earth and Environmental Science*. Bd. 268. IOP Publishing. 2019. DOI: 10.1088/1755-1315/268/1/012025.

[Tol+21] Ilya O Tolstikhin u. a. „MLP-Mixer: An all-MLP Architecture for Vision". In: *Advances in Neural Information Processing Systems*. Bd. 34. Curran Associates, Inc., 2021, S. 24261–24272.

[Tur+16] Hjalmar K Turesson, Sidarta Ribeiro, Danillo R Pereira, Joao P Papa und Victor Hugo C de Albuquerque. „Machine learning algorithms for automatic classification of marmoset vocalizations". In: *PloS one* 11.9 (2016), e0163041. DOI: 10.1371/journal.pone.0163041.

[Tzi20] Panagiotis Tzirakis. „End2You: Multimodal Profiling by End-to-End Learning and Applications". In: *Proceedings of the 1st International on Multimodal Sentiment Analysis in Real-Life Media Challenge and Workshop*. MuSe'20. New York, NY, USA: Association for Computing Machinery, 2020, S. 9. DOI: 10.1145/3423327.3423513.

[VA06] Sergei Vassilvitskii und David Arthur. „k-means++: The advantages of careful seeding". In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 2006, S. 1027–1035.

[Van+22] Astrid Vanneste, Simon Vanneste, Kevin Mets, Tom De Schepper, Siegfried Mercelis, Steven Latré und Peter Hellinckx. „An Analysis of Discretization Methods for Communication Learning with Multi-Agent Reinforcement Learning". In: *arXiv preprint arXiv:2204.05669* (2022).

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser und Illia Polosukhin. „Attention is All you Need“. In: *Advances in Neural Information Processing Systems*. Bd. 30. Curran Associates, Inc., 2017.

[VBV22] Sergey Verbitskiy, Vladimir Berikov und Viacheslav Vyshegorodtsev. „Eranns: Efficient residual audio neural networks for audio pattern recognition“. In: *Pattern Recognition Letters* (2022). DOI: 10.1016/j.patrec.2022.07.012.

[Von92] Karl Von Frisch. „Decoding the Language of the Bee“. In: *Nobel Lectures* (1992), S. 76. DOI: 10.1515/9783110253436.141.

[Wan+17] Feng Wang, Xiang Xiang, Jian Cheng und Alan Loddon Yuille. „NormFace: L2 Hypersphere Embedding for Face Verification“. In: *Proceedings of the 25th ACM International Conference on Multimedia*. MM '17. New York, NY, USA: Association for Computing Machinery, 2017, S. 1041–1049. DOI: 10.1145/3123266.3123359.

[Wan+20] Kaixin Wang, Bingyi Kang, Jie Shao und Jiashi Feng. „Improving Generalization in Reinforcement Learning with Mixture Regularization“. In: *Advances in Neural Information Processing Systems*. Bd. 33. 2020, S. 7968–7978.

[Wan+21] X. Wang, Y. Li, H. Zhang und Y. Shan. „Towards Real-World Blind Face Restoration with Generative Facial Prior“. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Juni 2021, S. 9164–9174. DOI: 10.1109/CVPR46437.2021.00905.

[Web15] World Wide Web. *Calculus on Computational Graphs: Backpropagation*. Online. http://colah.github.io/posts/2015-08-Backprop/, letzter Abruf: 14.07.2022. 2015.

[Web20] World Wide Web. *Leaking Pipes*. Online. https://discoverwater.co.uk/leaking-pipes, letzter Abruf: 14.07.2022. 2020.

[Wen+13] Felix Weninger, Florian Eyben, Björn W Schuller, Marcello Mortillaro und Klaus R Scherer. „On the acoustics of emotion in audio: what speech, music, and sound have in common“. In: *Frontiers in psychology* 4 (2013), S. 292. DOI: 10.3389/fpsyg.2013.00292.

[Wre+17] Peter H Wrege, Elizabeth D Rowland, Sara Keen und Yu Shiu. „Acoustic monitoring for conservation in tropical forests: examples from forest elephants“. In: *Methods in Ecology and Evolution* 8.10 (2017), S. 1292–1301. DOI: 10.1111/2041-210X.12730.

[WS22] Yutong Wang und Guillaume Sartoretti. „FCMNet: Full Communication Memory Net for Team-Level Cooperation in Multi-Agent Systems“. In: *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*. AAMAS '22. Richland, SC: International Foundation for Autonomous Agents und Multiagent Systems, 2022, S. 1355–1363.

[XHF21] Annie Xie, James Harrison und Chelsea Finn. „Deep Reinforcement Learning amidst Continual Structured Non-Stationarity“. In: *Proceedings of the 38th International Conference on Machine Learning*. Bd. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, S. 11393–11403.

[Yin+21] Yanling Yin, Ding Tu, Weizheng Shen und Jun Bao. „Recognition of sick pig cough sounds based on convolutional neural network in field situations“. In: *Information Processing in Agriculture* 8.3 (2021), S. 369–379. DOI: 10.1016/j.inpa.2020.11.001.

[Yu+21] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen und Yi Wu. „The surprising effectiveness of ppo in cooperative, multi-agent games“. In: *arXiv preprint arXiv:2103.01955* (2021).

[Zbo+21] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun und Stephane Deny. „Barlow Twins: Self-Supervised Learning via Redundancy Reduction“. In: *Proceedings of the 38th International Conference on Machine Learning*. Bd. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, S. 12310–12320.

[Zha+18a] Chiyuan Zhang, Oriol Vinyals, Remi Munos und Samy Bengio. „A study on overfitting in deep reinforcement learning“. In: *arXiv preprint:1804.06893* (2018).

[Zha+18b] Ziping Zhao, Yiqin Zhao, Zhongtian Bao, Haishuai Wang, Zixing Zhang und Chao Li. „Deep spectrum feature representations for speech emotion recognition“. In: *Proceedings of the Joint Workshop of the 4th Workshop on Affective Social Multimedia Computing and first Multi-Modal Affective Computing of Large-Scale Multimedia Data*. 2018, S. 27–33. DOI: 10.1145/3267935.3267948.

[Zha+20a] Amy Zhang, Clare Lyle, Shagun Sodhani, Angelos Filos, Marta Kwiatkowska, Joelle Pineau, Yarin Gal und Doina Precup. „Invariant Causal Prediction for Block MDPs“. In: *Proceedings of the 37th International Conference on Machine Learning*. Bd. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, S. 11214–11224.

[Zha+20b] Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal und Sergey Levine. „Learning Invariant Representations for Reinforcement Learning without Reconstruction“. In: *International Conference on Learning Representations (ICLR)*. 2020.

[Zha+21a] Hongming Zhang, Ke Sun, Bo Xu, Linglong Kong und Martin Müller. „A Simple Unified Framework for Anomaly Detection in Deep Reinforcement Learning“. In: *arXiv:2109.09889* (2021).

[Zha+21b] Yu Zhang, Peter Tiňo, Aleš Leonardis und Ke Tang. „A Survey on Neural Network Interpretability“. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 5.5 (2021), S. 726–742. DOI: 10.1109/TETCI.2021.3100641.

[ZT19] Rui Zhao und Volker Tresp. „Curiosity-driven experience prioritization via density estimation“. In: *arXiv:1902.08039* (2019).

[ZW18] Andrew Zhai und Hao-Yu Wu. „Classification is a strong baseline for deep metric learning“. In: *arXiv preprint arXiv:1811.12649* (2018).

[Zwe+21] Joeri A. Zwerts, Jelle Treep, Casper S. Kaandorp, Floor Meewis, Amparo C. Koot und Heysem Kaya. „Introducing a Central African Primate Vocalisation Dataset for Automated Species Classification“. In: *Proc. Interspeech 2021*. 2021, S. 466–470. DOI: 10.21437/Interspeech.2021-154.