Lydia Gauerhof

Essential properties for safe behaviour of a perception function in automated driving

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

Lydia Gauerhof

Essential properties for safe behaviour of a perception function in automated driving

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

1. Gutachter: Prof. Dr. Dirk Beyer
2. Gutachter: Prof. Dr. John Alexander McDermid
3. Gutachter: Prof. Dr. Eyke Hüllermeier

Tag der Einreichung: 27. Oktober 2022
Tag der mündlichen Prüfung: 19. September 2023

Eidesstattliche Versicherung

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

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

Gauerhof, Lydia
____
Name, Vorname

Sindelfingen, den 20.02.2024
____
Ort, Datum

Zusammenfassung

Der jüngste Erfolg des Maschinellen Lernens (ML) hat zu einer weiten Verbreitung in verschiedenen Bereichen geführt, z. B. bei Haushaltsprodukten, bei der Verarbeitung natürlicher Sprache und bei Empfehlungsprogrammen. Technologische und rechnerische Fortschritte sowie die zunehmende Verfügbarkeit von Daten haben diesen Trend begünstigt. In diesem Zusammenhang wird erwartet, dass ML die Sicherheit von autonomen, sicherheitskritischen Systemen aufgrund seiner überzeugenden Leistungsfähigkeit erhöhen wird.

In einer Sicherheitsargumentation soll aufgezeigt werden, dass eine Funktion ausreichend sicher ist, also ein akzeptables Restrisiko nicht überschritten wird. Dazu soll der Nachweis erbracht werden, dass bestimmte Eigenschaften, die die Sicherheit gewährleisten, erfüllt sind und Fehlerursachen angemessen mitigiert werden. Aus diesem Grund besteht das Ziel dieser Arbeit darin, Eigenschaften der ML-basierten Funktion und der Daten, die für das Sicherheitsargument wesentlich sind, zu extrahieren und zu bestätigen. Die Arbeit befasst sich also mit der Forschungsfrage, welche Eigenschaften für ein sicheres Verhalten einer ML-basierten Wahrnehmungsfunktion notwendig sind und wie wir sie erwerben können.

Dabei ist der Einsatz von ML in sicherheitskritischen Systemen, wie z. B. Wahrnehmungsfunktionen beim automatisierten Fahren, mit der Herausforderung verbunden, ein überzeugendes Sicherheitsargument zu liefern. So verarbeiten ML-basierte Wahrnehmungsfunktionen Sensordaten und extrahieren Informationen, z. B. Objekte und befahrbare Bereiche. Wenn die Wahrnehmung versagt, kann es zu großen Schäden und Todesfällen kommen. Daher muss in komplexen Umgebungen das Risiko eines Versagens von ML-Funktionen auf ein akzeptables Niveau gesenkt werden. Etablierte Normen in der Automobilbranche wie die funktionale Sicherheit ISO 26262 und die Sicherheit der intendierten Funktionalität ISO 21448 geben nicht ausdrücklich an, wie mit ML umzugehen ist. Zudem können etablierte Ansätze, die in den Normen empfohlen werden, nicht direkt auf ML-Funktionen angewendet werden. Aus diesem Grund arbeiten wir in dieser Arbeit zunächst die Herausforderungen bei der Validierung von Wahrnehmungsfunktionen heraus [44].

Einerseits ist der Eingaberaum eines automatisierten Systems und die beabsichtigte Funktionalität sehr komplex. Andererseits ist die Formulierung der Spezifikation einer Wahrnehmungsfunktion eine Herausforderung für sich. Darüber hinaus ist es schwierig, die geforderten Eigenschaften für den gesamten Eingaberaum zu implementieren und die implementierte Funktion in Bezug auf die beabsichtigte Funktionalität zu verifizieren und zu validieren.

Unser Fokus liegt auf der Spezifikation der notwendigen Eigenschaften einer Wahrnehmungsfunktion für automatisierte, sicherheitskritische Systeme und deren Realisierung, vom Entwurf bis zur Verifizierung und Validierung. Nur beides zusammen, Spezifikation und Realisierung, ermöglichen es, ein umfassendes Sicherheitsargument zu entwickeln. Konkret wird der Anwendungsfall einer sicherheitskritischen Fußgängererkennungsfunktion für das automatisierte Fahren untersucht. Zu diesem Zweck wird ein “Deep Neural Network” (DNN) zur Fußgängererkennung trainiert und auf seine Eigenschaften hin untersucht. Darüber hinaus werden neuartige Methoden entwickelt [41, 42, 96], um diese Anforderung zu erfüllen.

Zunächst wird eine Reihe von Sicherheitsanforderungen an die Wahrnehmungsfunktion abgeleitet und auf ihre Auswirkungen auf die Aktivitäten im ML-Lebenszyklus

untersucht [43]. Zusätzlich werden funktionale Unzulänglichkeiten untersucht, weil sie zu Gefahren führen können [42]. Zu diesem Zweck werden die relevanten Datencharakteristiken extrahiert. Es werden Fehlerkategorien identifiziert und Abhilfemaßnahmen vorgeschlagen, wobei der Schwerpunkt auf der Eignung der Trainingsdaten liegt. Zusätzlich zu dem Ansatz, die Trainingsdaten zu verbessern, werden weitere Maßnahmen getroffen. Es wird der Fall, bei dem sich die Daten sehr stark von den bereits eingesetzten Daten unterscheiden, analysiert werden. Dabei schlagen wir vor, die Dateneignung durch eine Online-Anomalieerkennung zu ergänzen, die das Verhalten des DNNs überwacht[42].

Hierzu stellen wir zwei neue Arbeiten zur Anomalieerkennung vor. Während FACER darauf trainiert ist, verschiedene Arten von Rauschen zu erkennen, die die Daten verzerren können [96], ist ReverseVAE in der Lage, Anomalien außerhalb der Verteilung zu erkennen [41]. Beide Anomalien können einen großen Einfluss auf das sichere Verhalten einer ML-Komponente haben. Eine weitere Fähigkeit von ReverseVAE [41] ist die Möglichkeit, die Daten mit bestimmten visuellen Attributen zu manipulieren. So können Daten mit definierten visuellen Attributen generiert werden, die später zum Beispiel zum Trainieren oder Testen der ML-basierten Funktion eingesetzt werden können.

Da das Testen einer ML-Funktion nicht nur durch die Spezifikation geleitet werden kann, werden unterschiedliche Ansätze zur Spezifikation eines Testorakels und Testansätze aus unterschiedlichen Domänen und Anwendungsgebieten analysiert und neue Test Setups entwickelt [2].

All die Herausvorderungen führen dazu, dass die Entwicklung, inkl. der Verfeinerung von Anforderungen in der Spezifikation, iterativ und damit kontinuierlich stattfindet. Um die Rückverfolgbarkeit zu kompensieren, die zwischen den Anforderungen und den Codezeilen bei einer ML-basierten Funktion fehlt, schlagen wir explizite Verbindungen zwischen den Artefakten vor und illustieren diese mit Beispielen [40].

Im Ergebnis bietet diese Arbeit eine ganzheitliche Sicht auf die Forschungsfrage, welche Eigenschaften für ein sicheres Verhalten einer ML-basierten Wahrnehmungsfunktion erforderlich sind und wie wir sie gewährleisten können. Damit soll die Lücke zwischen bereits etablierten Sicherheitspraktiken und wissenschaftlichen Erkenntnissen in der ML-Entwicklung geschlossen werden.

Abstract

The recent success of Machine Learning (ML) has led to the widespread application of ML in various domains, such as household products, natural language processing, and recommendation programs. Technological and computational advances and the increasing availability of data have fueled this trend. In this context, machine learning is expected to enhance the safety of autonomous safety-critical systems due to its compelling performance.

In a safety case, it shall be shown that a function is sufficiently safe, i.e., an acceptable residual risk is not exceeded. To this end, it should be demonstrated that certain properties that ensure safety are satisfied and that causes of failures are adequately mitigated. For this reason, the goal of the work is to extract and confirm properties of the ML-based function and the data that are essential to the safety argument. Thus, the work addresses the research question, what properties are necessary for a safe behaviour of an ML-based perception function, and how can we acquire them.

Thereby, the use of ML in safety-critical systems, such as perception functions in automated driving, comes with the challenge of providing a convincing safety argument. For example, ML-based perception functions process sensor data and extract information about objects and drivable areas. When perception fails, major damage and fatalities can result. Therefore, in complex environments that evolve over time, the risk of ML functions failing must be reduced to an acceptable level. Established standards in the automotive industry such as functional safety ISO 26262 and safety of intended functionality ISO 21448 do not explicitly state how to ensure the safety of ML-based functions. In addition, established approaches recommended in the standards cannot be directly applied to ML components. For this reason, we elaborate the challenges in validating perception functions [44].

On the one hand, the input space of an automated system and the intended functionality is very complex. On the other hand, formulating the specification of a perception function is a challenge in itself. Moreover, it is difficult to ensure the required properties hold over the entire input space and to verify and validate the implemented function with respect to the intended functionality.

Our focus is on the specification of the essential properties of a perception function for automated safety-critical systems and its realization, starting with design up to verification and validation. Only the two together, specification and realization, make it possible to develop a comprehensive safety argument. Specifically, the use case of a safety-critical pedestrian detection function for automated driving is investigated. For this purpose, a Deep Neural Network (DNN) for pedestrian detection is trained and its properties are investigated [42, 43]. In addition, novel methods are developed [41, 42, 96] to satisfy its essential properties.

First, a set of safety requirements is derived and examined for their impact on the activities of the ML lifecycle [43]. In addition, functional insufficiencies are investigated as they might lead to hazards [42]. To this end, relevant data characteristics are extracted. Error categories are identified and remedial actions are proposed, focusing on the suitability of the training data. In addition to the approach to improve the training data, other measures are taken. When input data strongly differs from that used in training and test, its impact on the performance should also be analysed. In this case, we propose to complement data suitability with online anomaly detection that monitors the behaviour of the DNN [42].

To this end, we present two recent publications on anomaly detection. While FACER is trained to detect different types of noise that can distort the data [96], ReverseVAE is able to detect anomalies outside the distribution of training data [41]. Both of these anomalies can have a large impact on the safe behaviour of an ML-based function. Another capability of ReverseVAE [41] is the ability to manipulate the data with certain visual attributes. Thus, data can be generated, with defined visual attributes, which could later be used for training or testing of an ML-based function.

Since testing of an ML function cannot be guided only by the specification alone, we present different approaches to specifying a test oracle and testing approaches from different domains and application areas and novel test setups developed [2].

In order to address the challenges outlined above, an iterative and continuous specification of requirements in interaction with the development is proposed. To compensate for the traceability that is missing between the requirements and the lines of code in an ML-based function, we propose explicit artefact links and illustrate this with examples [40].

All in all, this work provides a holistic view on the research question of what properties are required for a safe behaviour of a ML-based perceptual function and how we can acquire them. This is intended to bridge the gap between already established safety practices applied to non ML-based systems and scientific knowledge in ML development.

Acknowledgements

There are a number of people to whom I owe a great debt of gratitude. Without them, I would never have gone the way I have.

My parents have given me so many things along the way that I cannot count or name. They have always supported me and would have traveled to the other side of the world if it had been necessary. Fortunately, they were spared that, but just thinking about it always reassures me - no matter how old I am.

Kilian, my partner and my love, you enrich my life in every respect. Your confidence and support have always encouraged me to keep going. We have already experienced so much together and shared so much in the past. All the more I look forward to all that is yet to come.

Dirk Beyer, my doctoral advisor, I would have not passed all the challenges in the last years without your experience and your support. I appreciate that very much and I am very thankful for all your efforts.

Special thanks also go to second reviewer John McDermid and third reviewer Eyke Hüllermeier.

Simon Burton, mentor and companion, you already brought me closer to topic of safety assurance a few years ago, when it was just one term of many that had come up in development. He thaught me that it is much easier to develop a function than to know during development when it may fail. I really appreciate his support, his experience in automotive safety and his enthusiasm to make things happen.

Colleagues from the University of York during the fellowship of the Assuring Autonomy International Programme, all of you, Chiara Picardi, Richard Hawkins, Colin Paterson, Ibrahim Habli, to be named first and also others, have me impacted heavily and sharped my knowledge about safety. The discussions with you were very valuable for me and I enjoyed working with you very much.

There are also colleagues and friends who have intentionally or unintentionally supported me with my dissertation. Simon Greiner, to be named first, with his motivating manner, Carmen Cârlan, Barbara Gallina, Hana Boukricha, Keerthana Govindaraj, Julia Leibinger, Maria Lyssenko, Yulia Svetashova, Ahmad Adee, Frédérik Blank, Volker Fischer, Roman Gansch, Christian Heinzemann, Andreas Heyl, Arut Kaleeswaran, Peter Munk, Arne Nordmann, Andreas Rohatschek, Christoph Schorn, Markus Schweizer, Andreas Albrecht, Matthias Woehrle.

Contents

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Contributions . . . 2
1.3 Outline . . . 4

2 Background **7**
2.1 Use Case . . . 7
2.2 Standards on Automotive Safety and ML . . . 9
2.3 Safety Cases . . . 11

3 Related Work **15**
3.1 From System to Component Level . . . 15
3.2 Realization of Essential Properties . . . 16

4 Specification of a Safe ML-based Perception Function (B.1–B.3) **19**
4.1 Safety Case for the Pedestrian Detection Function . . . 19
4.2 Safety Requirements of an ML-based Perception Function . . . 20
4.3 Triggering Conditions . . . 21

5 Realization of a Safe ML-based Perception Function (B.4–B.7) **23**
5.1 Anomaly Detection . . . 23
5.2 Anomaly detection and Data Augmentation . . . 24
5.3 Testing . . . 25
5.4 Traceability of Assurance Artefacts . . . 26

6 Conclusion and Future Research **27**
6.1 Conclusion . . . 27
6.2 Future Research . . . 28

Bibliography **31**

A Credits **45**

B Original Manuscripts **47**

List of Figures

1.1 Overview of the articles in Chapter B and their placement in the ML lifecycle. 3

2.1 Data sample with predicted bounding boxes of an ML-based pedestrian detection. 8
2.2 Basic example of a Goal Structuring Notion (GSN) 12

If cars were like computers...
"For no reason at all, your car would crash twice a day."
- General Motors press release -

1 Introduction

In this chapter, we start with the motivation in Section 1.1 that established safety standards are not suitable for ML-based perception functions. Then we present the scientific contributions in Section 1.2. Finally, we show the outline of our work in Section 1.3.

1.1 Motivation

A crossing truck with white tarpaulin was misclassified by a perception function in Tesla's Autopilot, leading to a fatal accident [70]. The driver was not steering and the car crashed into the truck. The driver did not survive. Of course, the Autopilot is not developed to be used as a self-driving car, an Automated Driving System (ADS), but as an assistance system. However, many users apply and rely on it, as if the Autopilot would be already an ADS. Further fatal accidents followed [8, 37]. According to Tesla Deaths [109], which provides a crowd-sourced list of accidents involving Tesla vehicles, 15 deaths through August 20, 2022 are related to Autopilot use. This is almost certainly an underestimate of the total number of fatalities.

Other fatal accidents from other providers also occurred where either an assisted driving function or an automated driving function was tested: In 2018, a pedestrian pushing her bike was run over by an Uber car, even though there was a safety backup driver in the Uber car who was supposed to prevent exactly such a situation [10, 28].

Such accidents and others have followed, as there are no established standards, guiding the development of a safe perception function in automated driving. UL4600 [87] claims to guide this development. However, it is not yet established and still under revision. Other standards focusing on Artificial Intelligence (AI) and its subcategory Machine Learning (ML), such as ISO/AWI PAS 8800 [54], are also still under development.

As the consensus of main academic and the industrial players on safe behaviour of automated systems including ML components applied to complex environments has not been reached [22], this thesis contributes to the necessary, corresponding discussion.

Furthermore, the verification of components that usually supports the claims in a safety argument, is not trivial for ML-based systems [77]. While it is also not trivial even with components handcrafted during coding, there are established standards and many years of experience. ML components are trained on data, e.g. encoding the learnt pattern e.g. with weights [46]. There are approaches providing a proof via formal verification [61, 112, 115]. However, a formal verification and a mathematical proof remains for perception functions challenging for many reasons [77].

One of these reasons is the difficulty to come up with a thorough specification for a perception function [91]. The environment that an ADS is meant to control is complex and changes in over time [116]. While the high environmental complexity might be reduced for particular Operational Domains (ODs), such as operation of a ADS on private terrain with specifically educated persons with slow ego vehicle velocity [88], other ODs, such as urban driving will remain complex. Also the changes over time might lead to a malfunction under unexpected conditions [79, 120].

Furthermore, improvements on developing of ML methods, e.g. anomaly detection [6, 95, 121], explainability [11, 25, 73], uncertainty estimation [62, 86, 92] are measured based on a benchmark improving the performance of a novel method in comparison to other similar methods [15, 50, 81, 85]. This assessment is not appropriate for a safety argument. The best method might be still not good enough and - vice versa - a function with lower average performance might be safe, if it meets the safety requirements.

In summary, both the established safety concepts do not sufficiently address ML-specific challenges and the approaches widely used in the ML community do not meet the safety requirements. It is therefore all the more important to build a bridge between the different fields, such as AI, safety, and data management and to raise awareness of the challenges of the respective fields.We are the first in providing contributions within this context. We show how to specify essential properties of ML functions that are required for the safe behaviour of perception functions in automated driving. We provide two novel approaches to anomaly detection as well as a publication on testing. Finally, we present novel work on traceability to provide the linkage between requirements and evidence that e.g. utilize the results from testing.

1.2 Contributions

In this thesis we focus on the following stages of the ML lifecycle: safety assurance scoping, requirements elicitation, data management, design and training, and verification and validation. Please note that the artefacts of the first two stages, safety assurance scoping and requirements elicitation, are not necessarily required for training an ML function. In order to develop a safety-critical ML function, they are indispensable, though.

Figure 1.1 shows how the articles are located within the stages of the ML lifecycle. Since assuring safety of an ML-based perception function cannot be done with one method alone, we have made contributions in all stages of the ML lifecycle for our holistic approach. Furthermore, we show within the last contribution, that and how the artefacts generated in each stage of the ML lifecycle have to be linked explicitly with each other. Based on this, the missing traceability between the requirements and the lines of code can be established. The explicit artefact links in turn support the validity of the safety case. Although the articles often influence other stages besides their placement, this diagram is intended to contribute to the general understanding of the relationships among the articles.

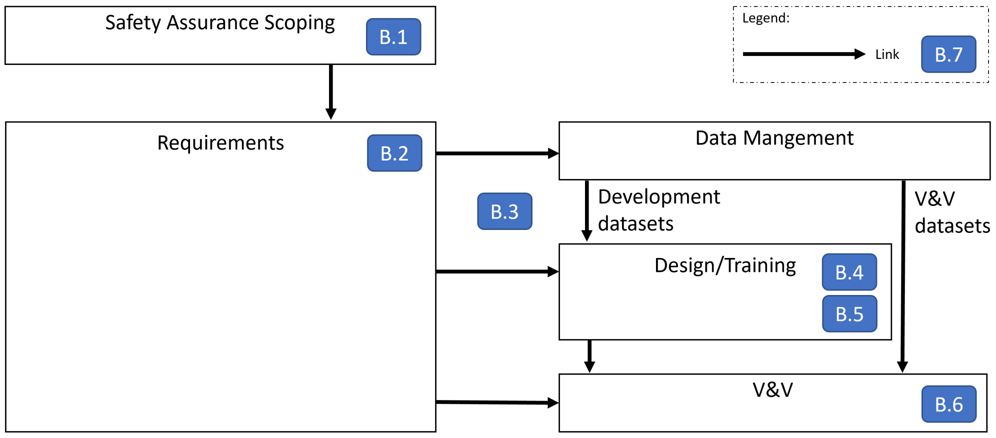


Figure 1.1: Overview of the articles in Chapter B and their placement in the ML lifecycle: While the article from Section B.1 contributes a first draft of a safety case for the pedestrian detection function, the article from Section B.2 is mainly in the stage of safety requirements driving the activities of the other stages. Since the article from Section B.3 covers the stages of requirements, data management and model training, its placeholder is between these stages. We assign the articles from Section B.4 and Section B.5 in design and training stage of the component, because they provide novel methods for anomaly detection and data augmentation. The article from B.6 is located in the stage of verification and validation. The article from Section B.7 concentrates on the traceability, so that its placeholder is set next to the link in the legend.

Our work comprises the following contributions:

- In Section B.1 we present validation targets to demonstrate which challenges must be overcome when specifying the safe behaviour of an ML function. Further, we provide first suggestions how to approach these challenges.
- In Section B.2 we present a method for generating safety requirements for an ML-based perception function of ADS. They are derived from the system level requirements of the ADS and embody the essential ML properties performance and robustness and the essential data properties accurateness, balance, relevance, completeness.
- In Section B.3 we provide a list of semantic categories for systematic errors of DNNs for pedestrian detection. In addition, we propose mitigation measures for the error categories by extending the training dataset with samples representing the error categories. Further, we introduce and evaluate an anomaly detection as a mitigation method.
- In Section B.4 we introduce the Feature Activation CheckER (FACER), an efficient and effective approach for the concurrent detection of multiple anomalies during the operation of DNNs. We evaluate detection performance and generalization abilities on eight noise types with different severities applied to images of three different datasets, CIFAR-10, CIFAR-100, and SVHN. We find that training on low severities of noise makes the anomaly detector generalize well to higher

severity levels. Furthermore, by combining multiple noise types during the training of FACER, we achieve high detection capabilities also on noise types not seen during training. We also evaluate the detection of unseen classes, meaning that the classes were not observed during training. FACER offers the advantage of being more flexibly applicable to other anomaly detection problems, while its detection performance is in the range of other state-of-the-art methods for this task.

- In Section B.5 we introduce the reconfigurable ML-based model Reverse Variational Autoencoder (Reverse-VAE) that it is used either for anomaly detection or for visual attribute manipulation as a data augmentation method to improve the DNN model robustness against anomalies. This means that the Reverse-VAE can not only learn an accurate mapping of high dimensional space to low dimensional space, but also generate realistic and diverse images (due to novel form of training setting). The good reconstruction performance of the Reverse-VAE is restricted on distribution of training data. In this context, every sample that is out of training distribution of Reverse-VAE is anomalous. Thereby, we assume that training distribution of Reverse-VAE is the same as from the ML-Component.
- In Section B.6, we analyse and evaluate existing work from different domains regarding their suitability for testing DNNs for visual perception in ADS. Thereby, we consider test input and test oracle generation as well as test adequacy separately. We conclude that testing of DNNs in this domain requires several diverse test sets. We show how such test sets can be constructed based on the presented approaches addressing different purposes and identify open research questions. By doing so, we present novel test setups to reduce safety concerns.
- In Section B.7 we show which links between the artefacts that are generated in different stages of the development must be established explicitly. Further, we demonstrate how that can be instantiated to provide traceability. These links enable us to build confidence in our safety argumentation. We concretize these explicit links in two examples, namely pedestrian detection and vehicle detection.

1.3 Outline

We structure this thesis as follows. Chapter 2 presents the foundations on which we have built our work. We start with an introduction to the use case in Section 2.1. Then we provide an overview of important standards in Section 2.2 which we divide into the subsection of safety standards in the automotive domain and the subsection of standards in the ML domain. Subsequently, we present safety cases in Section 2.3.

Chapter 3 provides an overview of the related work of research fields that also aim to assure the safety of ML components in automated systems, as each of our publication contains the corresponding related work itself. We start with approaches that strengthen the link between system and component level. We complete this section with an overview of further methods that are adjacent to our work and have the potential to provide evidence of essential properties.

In Chapter 4 and in Chapter 5 we present a summary of our contributions, which can be read in full detail in the respective original articles in the appended Chapter B. Each of these is introduced with a brief description of the underlying problem. While Chapter 4 is mainly dedicated to the specification, Chapter 5 covers the work on the realization of the essential properties.

Finally, we give a conclusion of our work and outlook on future research in Chapter 6. Chapter A presents the credits and Chapter B contains the original print versions of the presented articles.

2 Background

In this section we provide background information for a better understanding of the work. First, we motivate our chosen use case in Section 2.1 and second, we present the most important standards for automotive safety and ML in Section 2.2. In addition, we introduce safety cases in Section 2.3.

2.1 Use Case

In this section we introduce the role of perception of the environment in automated systems, such as ADS. Then we provide an overview of approaches for camera-based perception. Finally, we present our implementation that is used in several publications attached to this work.

Failure in the Perception of the Environment

An ADS perceives its environment via various sensors, such as camera, ultra sonic, radar, etc., whose data are processed [107]. Among other information, dynamic objects such as other road users must be identified, but also all the information needed for the driving task. This includes, for example, self-localization [59], recognition of drivable area [103] and road signs [106].

In contrast to human drivers, automated vehicles have so far only been able to extract missing information from the context to a limited extent [113, 114]. Humans can draw conclusions very quickly in unknown situations based on their experience and knowledge in order to drive safely [90]. This ability to generalize, to adapt and to be aware of risk is of the greatest value [64]. For example, human drivers can anticipate a lane change without a turn signal more than two seconds before before a vehicle in front leaves the lane [34].

On the other hand, most accidents are primarily due to human error, especially due to inadequate perception of the environment and distraction. The inadequate perception, for example when changing lanes, results from the complex competing goals of perceiving the front and rear lanes and observing the neighboring lane [36]. The likelihood for safety-critical events increases while being distracted by other tasks than driving, such as texting message on cell phone or rummaging through a grocery bag [84]. According to the Federal Statistical Office of the Federal Republic of Germany [117], human error

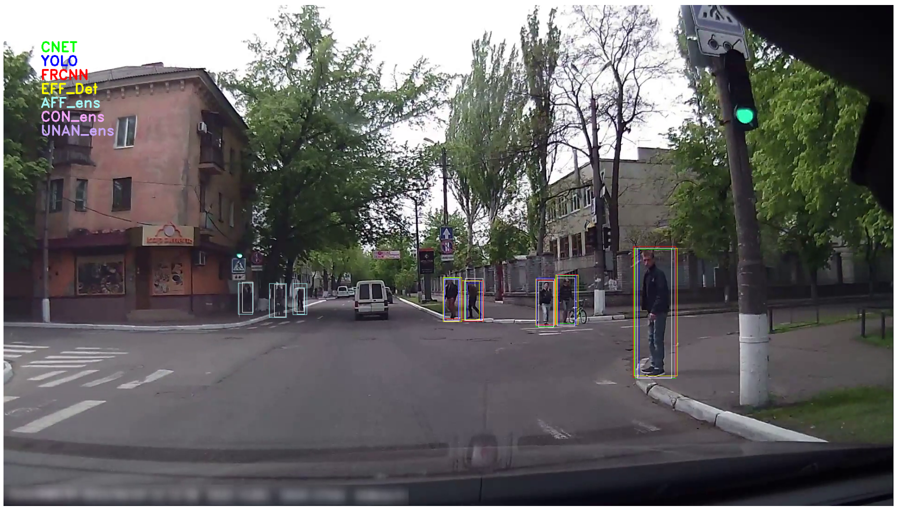


Figure 2.1: Data sample with predicted bounding boxes of an ML-based pedestrian detection.

was by far the most common cause of accidents: 88.0% of the causes in accidents with personal injury in Germany in 2021 were driver error and 2.7% pedestrian error.

This is also where the potential of ADSs is hidden: Vehicle automation is expected to reduce the accident rate if a high level of confidence in signal processing can be achieved [118]. In this context, guarantees can be made that the signal processing path identifies dangerous situations [118]. Thus, automated driving is not only about increasing efficiency, reducing the number of human drivers or even substituting them, but also about increasing road safety [79].

However, it should also be noted that the introduction of ADSs also means that automation failures will occur that would not be expected with human drivers [105]. The perception of an ADS in particular can make significant contributions to increasing, but also reducing, the number of traffic accidents. This applies in particular to the detection of vulnerable road users such as pedestrians.

If pedestrians are not detected by an ADS, also called False Negatives (FN), this can lead to human injury under certain circumstances. Therefore, pedestrians shall be prevented from being overlooked. If braking is applied to a falsely detected object that does not actually exist, a so-called ghost object or False Positive (FP), a sudden braking maneuver can lead to a rear-end collision [105].

Camera-based Perception

Images provided by a camera installed in the vehicle are processed to make sense of complex input data in the current scenario of the environment. Thereby, the scenarios are very diverse [124]. Especially, ML is used to recognize objects, such as vehicles, pedestrians, as ML outperforms non-learning algorithms [68]. ML-based functions learn to map inputs to outputs, by recognizing patterns existing in the data. In particular, Deep Learning (DL) [46], a subcategory of ML, is an important driver for perception tasks. DL is based on Deep Neural Networks (DNNs) and for perception functions DNNs are mostly trained supervised using labeled data.

Perception tasks are object detection [5, 125], semantic segmentation [30, 39, 48], instance segmentation [16, 47], to name only few. For example, object detection is conducted when a DNN predicts bounding boxes to localise a particular classes in an image [123]. Fig. 2.1 demonstrates an example of pedestrian detection from the Joint Attention for Autonomous Driving (JAAD) dataset [66] where predicted Bounding Boxes (BB) of different DNNs are shown. A DNN trained for semantic segmentation predicts for each pixel of the image a class [48].

Implementation of a Perception Function

We describe here the DNN that is implemented and the dataset that is used for our use case of a pedestrian detection in automated driving.

DNN: The ML component used for pedestrian detection in several publications that are attached to this work is a DNN, which consists of a Squeezenet and Region Proposal Network (RPN) [**SafeComp2020**, 42]. The architecture is comparably small, so that it can be implemented with low computation demand on dedicated hardware in an ADS. Pedestrian detection is a safety-critical function that, if it does not function as intended, can lead to serious damage.

Used Dataset: The DNN is trained and evaluated on the JAAD dataset [66]. The JAAD dataset includes 346 video clips recorded over 240 hours with approximately 82000 image samples while driving in America, Canada, Germany and Ukraine. These data includes various driving situations with a variation of all seasons, weather conditions (e.g., rain, snow and fog), and day and night time.

2.2 Standards on Automotive Safety and ML

As perception functions in automated driving are safety-critical, we discuss in this section safety standards that are established in automotive domain and how they refer to ML as well standards that focus on ML and their relation to safety.

Safety in Automotive Domain

Avizienis et al. [9] and their preceding work [71] provide well established generalized basic concepts and terminology of dependable computing. There safety is defined as “the absence of catastrophic consequences for the user(s) and the environment” [9]. Faults, errors, and failures pose a threat to the dependability of the system and must be handled accordingly. A *failure* is an event that occurs when the system does not provide the correct service. The system is subsequently in a sequence of external states of the system at least one of which deviates from the correct state. The deviation from the correct state is defined as an *error*. The term *fault* is used to denote an adjudged or hypothesized cause for an error. A fault can occur internal or external of a system.

Our goal is to show that the ML-based perception function is safe. For this purpose, the industry-specific safety standards must be complied with. In this section, we give an overview of the standards and at the same time show in which respects they provide insufficient guidance for ML-based functions.

The current main automotive safety standards, ISO 26262 and ISO 21448, complement each other and have been introduced mainly for assisted, manually-driven vehicles. The standard draft ISO/AWI PAS 8800 is planned to cover both, ISO 26262 and ISO 21448 while mainly focusing on the development and operation of ML components. Further, UL 4600 focuses specifically on automated vehicles.

ISO 26262 [58] focuses on the functional safety of electric and electronic systems used in road vehicles and has been established for many years in the automotive domain. Automated vehicles with higher automation level than *level* 2 on the SAE standard J3016 [108] are expected to comply with ISO 26262. However, this standard fails to provide guidance on the treatment of ML components. Relevant deficiencies of ISO 26262 with respect to ML are discussed by Salay et al. [94] and by Henriksson [51], for example. Besides that, it is complemented by Safety of the Intended Functionality (SOTIF) addressed in ISO 21448 which is the focus of this thesis.

ISO 21448 SOTIF [53] refers to the absence of unreasonable risk due to hazards resulting from functional insufficiencies of the intended functionality or by reasonably foreseeable misuse. Functional insufficiencies might result from inappropriate specification of the functionality on system level or from technical limitations in the implementation of the system. In this context, functional insufficiencies are caused by triggering conditions in combination with performance limitations. Triggering conditions are particular conditions of the environment leading to a system respond with hazardous behaviour. SOTIF appendix D provides a first guidance on the treatment of ML components. Appendix D states that "ML is mainly used when a full specification of the problem at hand is not possible (e.g. it is impossible to specify the data representation of a pedestrian in all varieties such that it could be always recognized by rule-based algorithm)" [53]. In addition, appendix D.2.3 of ISO 21448 indicates that the ODD and data sufficiency have an important role in the SOTIF activities. Both are concerned with the identification of triggering conditions that uncover limitations of the functionality. Our work contributes primarily to SOTIF.

ISO/AWI PAS 8800 [54], with the focus on safety and AI, targets the utilization of ML in road vehicles. It defines a set of safety-related properties that are compatible with existing approaches currently used in the standards ISO 26262 and ISO 21448. However, it is still under development.

UL 4600 [87] is, in contrast to ISO 26262 and ISO 21448, developed specifically to guide the safety assurance of automated and connected vehicles. UL 4600 requires the ADS to perform safely and as intended without human interaction and places the safety case as the most important working product. ADSs include partial driving automation on level 3 and higher levels of driving automation [108], where a human driver is not steering, breaking or accelerating besides during required situations. In this case, the human only has to take control of the vehicle within a defined transition period. ML is one of many components referenced in the standard that shall function safely, though.

Standards on AI and ML

At the time of writing this work, the main standards in the context of AI and ML are still in the formation phase.

ISO/IEC DTR 5469 [57] belongs to the AI community and is a technical report that discusses functional safety and AI in a generic sense and is not focused on automotive applications. This document is also still under development.

ISO/IEC 22989 [55] provides AI concepts and terminology which is helpful, but does not give guidance on the safety assurance of ML.

ISO/IEC 23053 [56] presents a framework describing the system components and their functions in the AI ecosystem.

Although both, ISO/IEC 22989 and ISO/IEC 23053, do not focus on safety-critical applications, they consider the topic of risk management that includes safety as part of the AI lifecycle. However, it does not contain sufficient guidance for ensuring the safety of a safety-critical system.

A wide-ranging presentation about AI standardisation landscape is found in the meta study conducted by the European Commission [29].

2.3 Safety Cases

In this section we explain what a safety case is and how it can be represented by graphical notations. Safety cases are not mandatory, but recommended in ISO 26262 [58] and ISO 21448 [53].

Safety Cases

A Safety Case (SC) is "a structured argument, supported by a body of evidence, that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given operating environment." [1, 80]. A safety case mainly consists of a safety argument and evidence while the safety argument forms the spine of the safety case showing how artefacts are related and combined to provide assurance of safety.

A safety argument focuses on the identification and mitigation of hazards associated with the system [49]. Therefore, it must show how the available evidence supports the overall claim that a component or a system is sufficiently safe. Safety arguments break down this claim into arguments that justify that an acceptable level of residual risk is achieved. This residual risk, in turn, is attained by identifying system hazards, their potential causes and measures for eliminating or mitigating the effects of the causes. Evidence, such as test results, development processes, etc. underpin such arguments [14]. The evidence provided to support the sub-claims as well as the safety argument are usually not perfect [49]. There is always some residual uncertainty in the validity of the argument or the completeness of the evidence, as the test coverage might be imperfect or the sub-claim on the selected test data samples might be questionable.

Safety goals usually are established by avoiding hazards at a system level [58]. For example, the hazard of unintended acceleration is specified in the safety goal to prevent unintended acceleration. During the design process safety goals are then decomposed with respect to the components in further safety requirements. In contrast to the safety goals, the refined safety requirements are allocated to components.

The SC must not only be subject to internal and external reviews by assessors, but could be used to establish a liability. While several automotive safety standards suggest to provide a SC, it is not mandatory for all of them, such as in ISO 26262 [58] and ISO 21448 [53]. If there is no SC, then the documentation of the component development is used in legal proceedings instead. However, developing a SC plays an significant role for the development of a safety-critical component that is used in automated systems. The safety load for ADS is particularly high compared to assisted, long-established systems where errors can also often be mitigated due to many years of experience in development and operation and by user involvement.

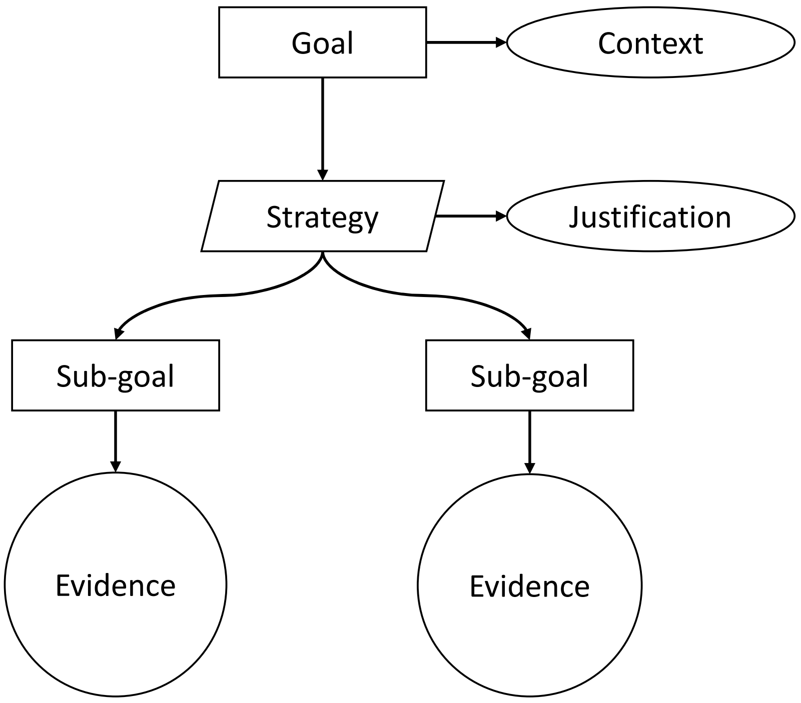


Figure 2.2: Basic example of a Goal Structuring Notion (GSN)

Graphical Notations for Safety Cases

Graphical representations, such as the Goal Structuring Notation (GSN) [1] depicted in Fig. 2.2 can be used to specify arguments. A graphical representation facilitates communication of the argument structure with colleagues, reviewers, and other authorities.

The standardized GSN[1] is a graphical argumentation notation that can be used to explicitly document the individual argumentation elements and artefacts. Claims are represented by goals and strategies indicate how goals are supported by other goals and finally by solutions representing evidence. Further, the context provides information that are needed to understand the corresponding goal and that are usually found in linked documents.

In consequence, a GSN-compliant SC includes a top-level goal that is decomposed via an argumentation strategy into subgoals determining the validity of the goal decomposition. The subgoals are either decomposed into further subgoals or supported by solution elements referring to the corresponding evidence. Context, reasoning, and assumption elements specify relevant contextual information usually provided by linked to artefacts. These artefacts are documents or products, such as the documentation of the ODD or the requirements, generated during the development.

Essential Properties

In this work, we consider essential properties for safe behaviour as properties that are necessary with respect to safety. These properties can also be referred to as safety-aware properties or safety-related properties.

These properties are usually embodied in the safety requirements, which are derived from the safety goals. We also consider ML-specific properties that cannot be associated at system level, although they also have a large impact on safe behaviour. For example, robustness against different sorts of noise, would not be derived from system level, as this depends on the implemented ML-function.

Since we focus on SOTIF, triggering conditions that can reveal systematic errors and thus, functional insufficiencies, are investigated. At the same time, special attention is paid to detection of and robustness to anomalies. We show different investigations and different measures to mitigate them. The presence of essential properties can be confirmed indirectly through the fulfillment of safety requirements or through extensive testing. In addition, the linking of essential artefacts in the safety argumentation ensures that these properties have been realized in a traceable way.

3 Related Work

Each article in Chapter B provides its corresponding related work separately. Here, we give an overview on other research fields that also aim to ensure the safety of ML components in automated systems. We divide them into (1) approaches to strengthen the link between system and component level and thus, the safety argument in Section 3.1 and (2) approaches to realize essential properties in Section 3.2. The latter are intended to provide evidence that supports the safety argument.

3.1 From System to Component Level

One shortcoming in safety case for ML is the linking argument that connects safety requirements at the system level to component requirements at the component level [93]. Thereby, only a valid decomposition of the safety requirements at the system level, so-called safety goals in automotive domain, into safety requirements at the component level justifies a high level of confidence in the safety of an ML component.

Salay et al. [93] propose a generic template on a deductive and formal approach to trace the requirements from system level to component level. In their formal claim decomposition, they identify a set of risk-aware safety metrics that can be used to evaluate perception components. These metrics, in turn, strengthen the link between requirements and Verification and Validation (V&V) during the development and assurance iterations. As an acceptance criteria, risk-aware safety metric, they use the misperception rate. In comparison to established performance metrics, this metric lends itself better to link the system level to the component level. This misperception rate is similar to the miss-rate that is used in our requirement RQ 1.1 [43].

Based on post-hoc segmentation of sub-objects, Schwalbe [98] proposes a consistent body part similarity of detected pedestrians and shows the invariance of internal representations of body parts with respect to the size in pixels of the depicted person. The findings that the representation of body parts is mostly size invariant which for example can be traced back to requirements that refer to body parts within the ODD. Due to the challenge of under-specification [43], relying only on concept embedding hides the risk of performance limitation. Data might include also occluded body parts that cannot be traced back to requirements or object parts are not well defined, so that the concept embedding would lead to an inappropriate functionality.

Lyssenko et al. [75] incorporate domain knowledge into the metric, so that the metric is used in a task-oriented way during the evaluation of an ADS pedestrian detection. To this end, the annotation of distance between the pedestrian and the ADS is essential, since close pedestrians are safety-relevant and a false detection likely becomes critical. They are able to adapt the data according to the extended data requirements, data acquisition specification and data labeling specification, since they use the CARLA simulator [23] to generate synthetic data. Thereby, the data set for training and test include pedestrians in a broad range of distances.

The Data Driven Engineering (DDE) process presented by Zhang et al. [122] links the operational design domain with the requirements and semi-automated generation of data sets. The process automation enables automated data set compilation. As the DDE offers traceability of all development artefacts including data sets and tools [122], it reflects the idea of continuous safety assurance. This provides the possibility to check data sets against well-defined data requirements and to detect gaps in data sets. This data analysis simplifies error analysis, especially with regard to errors caused by certain under- or over-represented data features. They demonstrate their approach for a use-case where data requirements are less complex than for pedestrian detection. Meaningful analysis in latent space for functions with high dimensional input space, e.g. in Lyssenko [75], Acar-Celik [3] and our work [43], might be challenging, as dimensions might be not clearly assignable. However, the DDE generally illustrates the importance of data in the safety assurance of ML-based components.

While safety analysis approaches for identifying potential failure modes in a system and their causes and effects, e.g. Fault Tree Analysis (FTA) within ISO 26262, are established safety methods, there is still a lack of experience on safety analysis approaches that can be applied to ML-based perception functions in complex environments. In this context, Thomas et al. [110] propose an approach to create useful safety analysis models for ADS perception functions. Their framework combines Event Sequence Diagrams (ESDs), FTA, and Bayesian Networks (BNs). To compensate this link between system and component level, they transform a BN into FTA utilizing the BN probabilities in FTAs. This framework is also capable to include evaluated basic events, such as triggering conditions, during the V&V phase.

3.2 Realization of Essential Properties

In this section we address selected methods that are adjacent to our work and have the potential to provide evidence of essential properties to support the safety argument. In particular, these include anomaly detection, explainability, and uncertainty estimation.

Anomalies are patterns in data that do not conform to a well-defined notion of normal behaviour [26]. Breitenstein et al. [18] provide a systematization of corner cases for visual perception in automated driving while referring to anomaly detection in computer vision as a well defined term used in different domains. Corner cases occur in many different forms and can be sample- or sequence related [17]. For example, there are collective anomalies that are instances and occur in unusually large quantities in an image. These collective anomalies can be detected by comparing the instance distribution of a test set (subset) with a training instance distribution (i.e. reference). Both distributions are obtained by instance-based semantic segmentation [17].

It is important for visual perception in automated driving to be able to reliably detect anomalies and corner cases during run-time [17] or if possible offline, during development. In this context, the investigation of triggering conditions [42] might fall under the definition of anomaly detection and corner case detection. If anomalies are well understood, it might be even possible to generate data with the particular anomalies to robustify the ML-based function [102]. In our work [42], we address the data suitability that leads to robustification. Furthermore, we complement data suitability with anomaly detection to compensate for the overconfidence of the ML-function and for the lack of robustness in the presence of certain anomalies.

Approaches on explainability and interpretability might make prior knowledge usable and solve the black-box problem of ML [19, 78]. Various approaches have been developed, such as Grad-Cam [100], to investigate which properties and features usually stored in the data are relevant to the model outcome [19]. Beckh et al. [11] present an overview and highlight open challenges, such as explanation quality that is often not addressed. Another open challenge is the transformation of intuitive knowledge as human understandable feedback to other e.g. formalized knowledge representations. These findings overlap with the challenge to document the leveraged knowledge in the safety argument properly, e.g. in form of ODD refinement or requirements refinement. We also encounter this challenge in requirements elicitation and have associated it with the semantic gap [44].

Furthermore, uncertainty approaches [7, 45, 97] might be investigated with respect to their potential to be included in the safety assurance [72], e.g. in relation to the correctness of the inference. Uncertainty is distinguished between aleatoric and epistemic uncertainty [52, 62]. Aleatory uncertainty is inherent in the data and refers to the inherent noise in the data, such as noise from the camera. This uncertainty cannot be reduced by adding more data to the model during training. In contrast, epistemic uncertainty addresses uncertainty due to lack of knowledge. It refers to the model parameters, i.e., the model-inherent uncertainty regarding the prediction. Therefore, epistemic uncertainty can be reduced by adding more data in training. In the development of such uncertainty methods, investigations are rarely performed that allow conclusions to be drawn about physical effects or particular data properties.

However, there are already uncertainty-based active learning approaches [81, 83, 101] for training data selection that improve the performance of the ML-based function. Thus, the uncertainty-based approaches contribute to data suitability and hence robustness. Unfortunately, hardly any research has been found that addresses whether uncertainty-based data selection improves the performance on data samples that are considered particularly difficult or reflect safety-critical situations. However, this and other approaches on data analysis, e.g. fuzzy sets in data analysis [31] could be an enrichment for the safety assurance of ML-based functions.

4 Specification of a Safe ML-based Perception Function (B.1–B.3)

In this section and in the next section we present each article that is appended in Chapter B with a short motivation and the main findings.

4.1 Safety Case for the Pedestrian Detection Function

The absence of unreasonable risk due to hazards caused by functional insufficiencies can only be achieved by a rigorous overall development approach - but what this entails for an ML-based perception function in automated driving is not covered by either the state-of-the-art or standards.

In the publication "Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving" (Section B.1), we present the first detailed safety case for a pedestrian detection supporting the argument of the absence of unreasonable risk due to hazards caused by functional insufficiencies by structuring the validation targets. We visualize our approach by a graphical notation GSN. We build upon our early work on making the safety case of ML in highly automated driving [20] and apply it focusing on the confidence in the safety argument [21].

Following the previously described challenges in ADS development - underspecification [65], semantic gap [12] and deductive gap [104] - we asses their implications for the safety-relevant function "pedestrian detection". Based on the ML-specific case study, we propose approaches to answer the following questions: (i) Underspecification: what is the intended functionality and what are its limitations? (ii) Semantic gap: How can the intended functionality be described? (iii) Deductive gap: How can requirements on the functional layer (here: ML) be implemented?

First, we introduce a pedestrian detection function of an ADS and the functional specification of this function. Then the known challenges of automated driving are investigated as causes of functional insufficicncies. Furthermore, we build our validation targets upon the functional insufficiencies, as they prevent inappropriate specification of the functionality on system level and technical limitations in the implementation of the system. By doing so, the causes of functional insufficiencies shall be mitigated. In addition, we provide concrete suggestions on which evidence might be valid to support the goals on the validation targets. Besides the validation targets of the ML-based function, not only functional modifications on functional level, but also on system level are discussed to achieve the intended functionality.

Reduction of risk due to hazards caused by underspecification. If the intended functionality is more diverse than it is specified [65], it may be underspecified. As a result, the defined use cases are only a subset of the intended functionality. To decrease the risk of underspecification, we suggest the following sub-goals of the validation targets: Environment is sufficiently well known; the task of the function is sufficiently well known; sensitivity against impact of environmental attributes is sufficiently low.

Reduction of risk due to hazards caused by semantic gap. The semantic gap occurs when implicit knowledge about the satisfaction of safety goals [12] is applied without documenting or justifying it. In the context of ML, the semantic gap may refer to statements about the relevance of references used for training, testing, and validation datasets. We propose the following sub-goals to underpin the goal to reduce the risk due to semantic gap: Pedestrian classes are sufficiently accurately described; location accuracy is sufficiently well described; the discrepancy between the real and described environments is sufficiently small.

Reduction of risk due to hazards caused by deductive gap. Deductive gap can occur when invalid assumptions are made at different levels of abstraction [104]. This can lead to unintended functionality. In the context of ML, features can be incorrectly learned or incorrectly implemented. The following sub-goals are proposed to be satisfied: Data set is sufficient for the intended functionality; overfitting is sufficiently reduced; underfitting is sufficiently reduced; essential influences on the ML function are sufficiently understood; ML function is sufficiently robust; learnt features are sufficient for function.

4.2 Safety Requirements of an ML-based Perception Function

As the environment of an ADS is very complex and changes over time, the derivation of the requirements for the perception function is not trivial.

While in Section B.1 we elaborate the objectives of the requirements specification process in terms of the subgoals in the GSN, we concretize the requirements for a specific scenario and show how they are met in the publication “Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings” (Section B.2). We focus on pedestrian detection at intersections and perform an evaluation using the publicly available JAAD dataset. In particular, we are the first to derive ML safety requirements for a concrete scenario based on essential properties and analyse how these requirements guide safety activities in the data management and model learning phases.

Model learning safety assurance process. We discuss the ML lifecycle that is divided into five stages: Requirements elicitation, data management, model learning, model verification and model deployment. For each of these stages we define essential properties, so-called desiderata. However, each function, e.g. pedestrian detection with bounding boxes, results in a different set of requirements that can be refined based on the desiderata.

Pedestrian detection at crossings scenario: safety requirements and their satisfaction. We specify first a safety goal that is a safety requirement at system level [53, 58]. Based on this requirement, we identify refined safety requirements that apply to the corresponding component, a pedestrian detection function in our use case. The definition of the ODD plays a crucial role here. Among the reasons to specify the ODD is to minimize the complexity of the input space and to document implicit assumptions on the input space.

To the best of our knowledge, we are the first that provided an traceable link between system safety requirements and ML safety requirements. First, we maintain the link with hazardous events at the vehicle level, and second, we ensure that safety aspects are considered in each stage of the ML lifecycle. We show that the ML safety requirements can guide and constrain safety assurance activities.

4.3 Triggering Conditions

Despite a thorough requirements elicitation, dangerous events might occur caused by performance limitations in combination with triggering conditions [53]. Thus, in addition to a thorough requirements elicitation, triggering conditions that occur specifically for each ML-based function shall be analysed. That is why we investigate how can triggering conditions be identified for an ML-based perception function and what are the implications for safety assurance.

In the publication “Considering Reliability of Deep Learning Function to Boost Data Suitability and Anomaly Detection” (Section B.3) we stress the increasing importance of the suitability of the training data for ML. First, we show how to extract the relevant data characteristics, such as relevant semantic features, and we identify error categories associated with triggering conditions. In addition, we propose to focus on data suitability. Complimentary to this, other measures should also be taken because not all data, such as out-of-distribution data, can be anticipated during development or accurately attributed to physical effects in reality. To deal with unknown out-of-distribution data that results also in triggering conditions we employ online anomaly detection with FACER, which monitors the behaviour of the DNN.

Analysis of performance limitations and known triggering conditions. Time-dependent, sequential data can change only to a limited extent from one sample to another due to physical constraints of the environment and can thus often be attributed to physical effects. This a-priori knowledge is used in the analysis of the triggering conditions and allows us to distinguish between errors caused by systematic semantic insuffciencies in the trained function and other effects, such as missing robustness. Therefore, we investigate whether there are recurring errors of the pedestrian detection (PDET) function in a sequence which results in functional insufficiencies. This allows us to extract learned correlations, some of which are not intuitively detected by looking at the data alone. We proceed in two steps. First, we extract FNs and FPs on pedestrian detections from video sequences. If the PDET function does not detect a pedestrian in the data sample, it is a FN. If a ghost is detected, it is a FP. In the second step, faulty sequences are checked for similarities in the data characteristics. If correlations can be detected between FP or FN and similar data characteristics, this may be a false learned correlation. This suggests weaknesses in the suitability of the training data. With the division of the errors into different error categories, we aim to gain a deeper understanding of the FPs and FNs to improve the training dataset. For the JAAD dataset and the PDET function in our case study, we identify ten error categories that can be assigned to known triggering conditions.

Unknown data - unknown triggering conditions. To simulate unknown triggering conditions we use unknown data to investigate the performance of the PDET function. In our case, we utilize random erasing. Two box types are erased from the data: rectangular boxes with and without noise, since they activate the feature detectors of the DNN in different ways. We then also provide a first solution to mitigate functional insufficiency. We introduce an anomaly detector that we train with different types of noise and evaluate its performance on different types of noise. We also investigate the performance on data that is not used in training in two different ways. We demonstrate the benefit of the anomaly detector for JAAD test data with random boxes added on the data and Cityscapes test data as out-of-distribution.

5 Realization of a Safe ML-based Perception Function (B.4–B.7)

In this chapter we present each article with a short motivation and the main findings.

5.1 Anomaly Detection

As robustness is one of the essential properties for safe behaviour of a perception function, robustness to anomalies must be ensured. To this end, DNN prediction performs best with data seen during training. These data are also called data within the distribution of the training data. When the input data is not part of the training distribution, it is called Out-Of-Distribution (OOD). OOD are among the various types of anomalies that can lead to incorrect inferences in ML based functions. Input data is affected by weather conditions, e.g. precipitation and lighting conditions. Sensors and other data processing components, such as filters, also manipulate the data. If some of these variations in the training data occur infrequently or not at all, they can eventually lead to being OOD.

In the publication "FACER: A Universal Framework for Detecting Anomalous Operation of Deep Neural Networks" (Section B.4), we introduce a novel approach to anomaly detection: the feature activation consistency checker (FACER). This consists of a very small DNN that receives intermediate results in the form of features of a main DNN as input. Thus, FACER forms a second head of the main DNN and outputs a binary result, anomalous or not. Since anomalies in the input lead to inconsistencies in the feature representation, FACER can detect them.

Our approach has several advantages. In contrast to other anomaly detectors that operate on the data samples directly, FACER processes intermediate results of the main DNN. Consequently, FACER takes into account the inconsistencies not only of the data samples, but of the learnt feature representation. As a result, the anomaly detection depends on the learnt patterns of the DNN with respect to the input data. In addition, FACER is trained independent from the main DNN that is used for the prediction task. Thus, FACER is applicable without retraining the main DNN. Different sets of weights for FACER can be used to detect different anomalies.

Evaluation of noise detection. We investigate the detection performance and generalization abilities of FACER on eight noise types with different severity levels applied to images from the CIFAR-10 [67], CIFAR-100 [67], and SVHN [82] tasks. CIFAR-10 includes colour images of 10 classes, such as cat and car, and the CIFAR-10 of 100 classes, respectively. SVHN includes colour images of street view house numbers. The results show that the anomaly detector can generalize well to higher severity levels by training it on low severity levels. In addition, FACER is able to generalize, as it achieves high detection performance even for noise types not seen during training.

Evaluation of unseen classes detection. Unseen class samples are also considered to belong to OOD and play an important role for safety. Our research shows that the detection performance of our method on unseen class samples is similar to other state-of-the-art methods. At the same time, it has the advantage of being more flexible for other anomaly detection problems than state-of-the art methods.

5.2 Anomaly detection and Data Augmentation

Variational Autoencoder (VAE) is one of the first models to target both image encoding, which allows data distribution analysis and image generation. Despite meaningful image embedding, VAEs tend to produce images with blurring effects, which limits its use in image generation and manipulation.

In the publication "Reverse Variational Autoencoder for Visual Attribute Manipulation and Anomaly Detection" (Section B.5), we introduce the Reverse Variational Autoencoder (Reverse-VAE), which can both learn an accurate mapping to low-dimensional space, the latent space, and generate realistic and diverse images. Reconfiguring our model allows us to detect anomalies and modify the images by manipulating attributes. Combining both applications enables us to detect anomalies and generate them in large quantities through attribute manipulation. This would ultimately reduce the training overhead for robust perception functions and make development more efficient. To this end, we use the recently proposed progressively growing strategy [60] to process high-resolution images with good scalability.

The approach. We minimize the Kullback-Leibler divergence [69] such that the novel form of training settings reduces the gap between the joint latent/data distribution of the generator and the joint distribution of the encoder.

The simple architecture results in our model being easily trained with fewer parameter settings. In addition, it can be upscaled using a PGGAN setting [60] to generate and reconstruct high-resolution images.

Applications. Both applications, the detection of anomalies and the subsequent generation of more realistic manifold samples of these anomalies, allow us to increase robustness to the corresponding anomalies. We show that the same model can implement both applications. The anomalous data contains a class that is not trained to be predicted and is therefore not part of the training data of the main model. The good reconstruction performance and good anomaly detection performance is achieved by being constrained to the distribution of the training data. For attribute manipulation, we train the model without auxiliary information, such as labeled attributes. After training, we extract dedicated visual attribute vectors in the latent space. For this, we use a small subset of labeled images. We can very flexibly edit desired attributes without having to re-train the model.

5.3 Testing

ML evaluations for benchmark purposes, purely statistical approaches based on splitting the data set, are not sufficient as testing approach for a safety-critical application, since essential properties and performance limitations remain unrevealed: Testing needs to be targeted towards multiple objectives, not just average performance in a single case. Most importantly, a complete specification including all essential properties is difficult to achieve in the open context of automated driving due to the complexity of the perception task.

In the publication "Testing Deep Learning-based Visual Perception for Automated Driving" (Section B.6), we mainly concentrate on the practical verification and especially on the functional testing of DNNs used for computer vision tasks in an automotive application. We consider approaches in terms of test input generation, test oracle and test adequacy that come from different domains such as software testing, ML, Computer Vision (CV), automated driving, and cyber-physical systems. We then discuss their applicability to the CV tasks in an automotive context. Combining different approaches has the potential to uncover properties or triggering conditions depending on the test objective.

Test input generation. Test data has a different purpose than training data. While training data is used to optimize the weights of the DNN to achieve the best possible overall performance, test data is intended to be able to highlight weaknesses in the DNN's functionality. After discussing local sampling techniques, we highlight the importance of both, domain and data analysis and present different approaches to conduct them. Finally, we discuss different ways of synthetic data generation, including augmentation techniques and their potential for improving testing. Based on these different data generation techniques, essential data properties can be embedded in the data, which in turn can be used for training a model or for testing essential properties.

Test oracle. Besides specification based oracles, we also present ground-truth based and derived oracles. Here it becomes clear that testing goes far beyond the specification to examine functionality. Besides specification-based oracles, there are also oracles from ground truth and meta information as well as derived oracles. They all have different advantages and disadvantages. While in the first case ground truth might include label noise [38] and human labeling errors [30], derived oracles might include metamorphic testing [99] requiring multiple implementations. These, in turn, might be wrong or results in less meaningful results. Nevertheless, it is essential to fulfill the specification - and in certain cases to adapt the specification, should testing reveal functional insufficiencies that can be traced back to an inappropriate specification.

Test adequacy. We use coverage to describe test adequacy which might be used for the test justification in the safety argument. This requires information from the system under test, e.g. for the structural coverage, or information from a mutation test as well as information from a model of the input domain.

5.4 Traceability of Assurance Artefacts

After elaborating the aforementioned aspects of a safe ML-based perception function, we focus on their implications for the safety case. For ML-based functions, there is no longer a direct mapping of the requirements to the code, since they are not programmed - created by hand - and the learned patterns are not explicit, but embedded, for example in weights. Traceability of the various artefacts generated for safety assurance is therefore a challenge and must be compensated for in other respects.

In the publication “On the Necessity of Explicit Artifact Links in Safety Assurance Cases for Machine Learning” (Section B.7), we present the essential artefacts that arise in the different phases of the ML lifecycle. In addition, we explain how they interact and how traceability can be facilitated based on explicit artefact links. These links allow us to build confidence in our safety argumentation. We use two examples, pedestrian detection [75] and vehicle detection [4], to illustrate these explicit links.

First use case: pedestrian detection. The first case study addresses the evaluation of pedestrian detection with task-oriented metrics [75]. The analysis shows that data management artefacts containing extracted information, data analysis, labeling, and annotations are of particular importance for traceability. The data management artefacts must be tightly linked to the requirements elicitation. One way to do so is to link the extracted information and data analysis with the semantic domain model, an extended version of the ODD. Another way is to provide a precise annotation specification, so that the annotation quality is high. The explicit links will allow us to trace the changes in the requirements directly affecting the data. By doing so, outdated or insufficient data can be traced and further measures taken, such as excluding invalid data.

Second use case: vehicle detection. The second case study addresses the systematic modeling of environmental perception constraints in the evaluation of automated driving [4]. Here it becomes apparent that the description of the input space, for example using a semantic domain model, is important to document the performance limitations and to trace their impact on data, requirements, and evaluation. The proposed explicit artefact links, used in early development and assurances phases, enables traceability in the lifecycle of an ML-based component. This forms the basis for the validity of the safety case, enables adaptations of autonomous systems in an evolving open context, and enables impact analysis.

6 Conclusion and Future Research

In this chapter we provide the conclusion of this thesis as well as ideas for future research that could follow-up on our work.

6.1 Conclusion

We provide a foundation for the safety assurance of ML-based perception functions in automated driving. To do so, we address the research question, what properties are necessary for a safe behaviour of an ML-based perception function, and how we can acquire them. We proceed as follows:

We provide a first detailed safety case for a pedestrian detection function in automated driving that is based on validation targets [44] (see also Section B.1). These form the objectives of the requirements specification process.

To provide a convincing safety argument for ML-based perception functions, we formulate meaningful safety requirements that guide the argumentation strategy [43] (see also Section B.2). By investigating their impact on the ML lifecycle, we provide a deep insight on consequences for activities artefacts of different stages of the ML lifecycle.

In addition to meeting safety requirements, we investigate functional insufficiencies that could compromise safety requirements [42] (see also Section B.3) to further refine the specification of the required safety-relevant properties. For this reason, we extract the relevant data characteristics. We define error categories, and propose mitigation measures, focusing on training data suitability. Despite all efforts to improve training data, not all possible variants of a real-world application can be identified. Therefore, we analyse the case of unknown, out-of-distribution data. In this case, we propose to complement data suitability with online anomaly detection using FACER, which monitors the behaviour of the DNN.

We present novel work on anomaly detection based on FACER [96] (see also Section B.4) and ReverseVAE [41] (see also Section B.5). While FACER is trained to detect different sorts of noise that can corrupt the data and unseen classes, ReverseVAE is able to detect out-of-distribution anomalies. All anomalies can have a large impact on the safe behaviour of a ML component. An additional ability of ReverseVAE is to be able to manipulate the data with specific visual attributes.

Since testing of an ML function might be not only guided by the specification of safe behaviour and the utilisation of safety-related metrics, we analyse and assess various approaches from different domains [2] (see also Section B.6). In addition, we provide different novel test setups and explain their test purpose. By doing so, the importance of specification is underlined as well as corresponding potential testing settings extracted.

All in all, these activities with their specific challenges have the consequence that an iterative development process is necessary, by which all artefacts - also the requirements - of the safety argument are affected. Traceability from the requirements to the lines of code does not exist for ML components because the requirements cannot be linked to specific portions of the architecture of an ML component. Instead, we require that the artefacts that are created in different stages of the ML lifecycle are linked to each other to ensure traceability [40] (see also Section B.7). We also illustrate these links with examples.

Our holistic approach provides a foundation for the safety assurance of ML-based perception functions in automated driving. We embody essential properties for a safe behaviour in the requirements and ensure essential properties in the ML-based function itself. Finally, we demonstrate that traceable artefacts are prerequisite for a convincing safety argument for a safe behaviour which is built upon the safety argument including the requirements and upon the evidence.

6.2 Future Research

In the following, we would like to point out individual issues that our work lays the groundwork for: acceptance criteria, change impact analysis, bridging the ML and safety community.

Acceptance criteria. It is challenging to come up with acceptance criteria on component level and the corresponding meaningful safety-aware metrics [27]. A vast variety of experiments and verification approaches are necessary, as there is not only the one property, such as e.g. adversarial robustness, to be proven [115]. Other properties, such as the ability to generalize [89], can be examined in terms of their suitability to support safe behaviour.

Performance metrics, such as ROC [35], F1-score [18], LAMR [33, 111], mAP [74], mIoU [18], evaluated on a given dataset that are established in the ML community might serve as a valuable benchmark. For a safety argument, statistical performance evaluation on one huge dataset is not appropriate, but should be extended with a specific analysis of the essential properties.

We have already pointed out, that the analysis of data sequences and its corresponding misperception rates [43] have great potential to be an essential safety-aware metric, as they play an important role in subsequent components, such as tracking. They could help to elevate the knowledge of the fault-error-failure relationship [9]. All these would lead to a better understanding of further triggering conditions and ease the development of mitigation measures and collection of relevant data. It might be possible to extract important test cases or to determine important training data [76]. Further analysis of relevant objects in the field of view of the ADS might be also an promising approach to shape safety-aware metrics [75]. Also the use of a safety-aware off-road people detection metric might be considered in safety requirements [119]. This metric could be based on the mean Average Precision (mAP) and include hazard risk correlated factors such as a person's distance to the vehicle and time-to-collision [119].

Moreover, there might be further investigations on stratified evaluations, shifted performance evaluations and contrastive evaluations [32]. In stratified evaluations [32], for example, we can focus on very specific data aspects only, e.g. particular weather condition, and provide evidence for particular required robustness, such as robustness against particular noise types. Leveraged knowledge and automated analysis of data characteristics would ease the identification and conformation of essential properties.

Change impact analysis. After an initially unknown triggering condition is evaluated, it should be added to the ODD. For pedestrians, for example, the pose may play an important role in the prediction quality. Thereby, an safety expert should check how the safety argument is affected by this change.

To this end, a change impact analysis can ease this task by guiding the safety expert with recommendations throughout the safety argument. To this end, we examine changes to artefacts and the dependencies between the artefacts throughout the safety argument on data sufficiency with the help of annotated change sensitive information [24]. The propagation of a change in an artefact is (semi-)automated with the help of qualitative and quantitative change impact analysis. Further changes in the safety argument should be traced and checked for impacts on other artefacts.

Other developments towards combination of model checking and data-flow analysis [13] could also be helpful.

Bridging the ML related and safety-related communities. Assuring the safety of a perception function in automated driving is heavily dependent on the state of the art. In addition, standards that incorporate the state of the art are necessary to provide a legal basis. The standards, in turn, require consensus in industry and science regarding the necessary claims and measures which for example is targeted in the publicly founded project "KI-Absicherung" [63]. Our work, such as [2, 20, 21, 40, 43, 44], contributed to the proposal of this project and to the project results. In addition, parts of the results are included in the standard draft ISO/AWI PAS 8800 [54] that will provide further legal guidance. The permeability of scientific and application-oriented findings in the individual fields of knowledge is a prerequisite for the targeted development of safe, automated systems.

Bibliography

[1] The Assurance Case Working Group (ACWG). *Goal Structuring Notation Standard*. The Assurance Case Working Group (ACWG). 2021. URL: https://scsc.uk/r141C:1?t=1.

[2] Stephanie Abrecht, Lydia Gauerhof, Christoph Gladisch, Konrad Groh, Christian Heinzemann, and Matthias Woehrle. "Testing Deep Learning-based Visual Perception for Automated Driving". In: *ACM Transactions on Cyber-Physical Systems (TCPS)* 5.4 (2021). ©2021 Copyright held by the owner/author(s). Publication rights licensed to ACM. DOI: 10.1145/3450356.

[3] Esra Acar Celik, Carmen Cârlan, Asim Abdulkhaleq, Fridolin Bauer, Martin Schels, and Henrik J. Putzer. "Application of STPA for the Elicitation of Safety Requirements for a Machine Learning-Based Perception Component in Automotive". In: *Computer Safety, Reliability, and Security*. Ed. by Mario Trapp, Francesca Saglietti, Marc Spisländer, and Friedemann Bitsch. Cham: Springer International Publishing, 2022, pp. 319–332. ISBN: 978-3-031-14835-4. DOI: 10.1007/978-3-031-14835-4_21.

[4] Ahmad Adee, Roman Gansch, and Peter Liggesmeyer. "Systematic modeling approach for environmental perception limitations in automated driving". In: *2021 17th European Dependable Computing Conference (EDCC)*. IEEE. 2021, pp. 103–110. DOI: 10.1109/EDCC53658.2021.00022.

[5] Yali Amit, Pedro Felzenszwalb, and Ross Girshick. "Object detection". In: *Computer Vision: A Reference Guide* (2020), pp. 1–9. DOI: 10.1007/978-3-030-03243-2_660-1.

[6] Jinwon An and Sungzoon Cho. *Variational autoencoder based anomaly detection using reconstruction probability*. 2015. URL: http://dm.snu.ac.kr/static/docs/TR/SNUDM-TR-2015-03.pdf (visited on 10/21/2022).

[7] Fabio Arnez, Huascar Espinoza, Ansgar Radermacher, and François Terrier. "A comparison of uncertainty estimation approaches in deep learning components for autonomous vehicle applications". In: 2020. DOI: https://doi.org/10.48550/arXiv.2006.15172.

[8] Ars Technica. *Autopilot was active when a Tesla crashed into a truck, killing driver*. URL: https://arstechnica.com/cars/2019/05/feds-autopilot-was-active-during-deadly-march-tesla-crash/ (visited on 08/20/2022).

[9] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. "Basic Concepts and Taxonomy of Dependable and Secure Computing". In: *IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING*. IEEE, 2004. DOI: 10.1109/TDSC.2004.2.

[10] BBC. *Uber's self-driving operator charged over fatal crash*. URL: https://www.bbc.com/news/technology-54175359 (visited on 08/20/2022).

[11] Katharina Beckh, Sebastian Müller, Matthias Jakobs, Vanessa Toborek, Hanxiao Tan, Raphael Fischer, Pascal Welke, Sebastian Houben, and Laura von Rueden. "Explainable Machine Learning with Prior Knowledge: An Overview". In: (2021). DOI: https://doi.org/10.48550/arXiv.2105.10172.

[12] Carl Bergenhem, Rolf Johansson, Andreas Söderberg, Jonas Nilsson, Jörgen Tryggvesson, Martin Törngren, and Stig Ursing. "How to reach complete safety requirement refinement for autonomous vehicles". In: *CARS 2015-Critical Automotive applications: Robustness & Safety*. 2015.

[13] Dirk Beyer, Sumit Gulwani, and David Schmidt. "Combining Model Checking and Data-Flow Analysis". In: *Handbook on Model Checking*. Ed. by E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem. Springer, 2018, pp. 493–540. ISBN: 978-3-319-10574-1. DOI: 10.1007/978-3-319-10575-8_16.

[14] Peter Bishop and Robin Bloomfield. "A Methodology for Safety Case Development". In: *Industrial Perspectives of Safety-critical Systems*. Ed. by Felix Redmill and Tom Anderson. London: Springer London, 1998, pp. 194–203. ISBN: 978-1-4471-1534-2. DOI: 10.1007/978-1-4471-1534-2_14.

[15] Hermann Blum, Paul-Edouard Sarlin, Juan Nieto, Roland Siegwart, and Cesar Cadena. "The Fishyscapes Benchmark: Measuring Blind Spots in Semantic Segmentation". In: *International Journal of Computer Vision* (2019). DOI: 10.1007/s11263-021-01511-6.

[16] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. "Yolact: Real-time instance segmentation". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 9157–9166. DOI: 10.1109/ICCV.2019.00925.

[17] Jasmin Breitenstein, Andreas Bär, Daniel Lipinski, and Tim Fingscheidt. "Detection of Collective Anomalies in Images for Automated Driving Using an Earth Mover's Deviation (EMDEV) Measure". In: *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*. 2021, pp. 90–97. DOI: 10.1109/IVWorkshops54471.2021.9669217.

[18] Jasmin Breitenstein, Jan-Aike Termöhlen, Daniel Lipinski, and Tim Fingscheidt. "Systematization of Corner Cases for Visual Perception in Automated Driving". In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. 2020, pp. 1257–1264. DOI: 10.1109/IV47402.2020.9304789.

[19] Nadia Burkart and Marco F Huber. "A survey on the explainability of supervised machine learning". In: *Journal of Artificial Intelligence Research* 70 (2021), pp. 245–317. DOI: 10.1613/jair.1.12228.

[20] Simon Burton, Lydia Gauerhof, and Christian Heinzemann. “Making the Case for Safety of Machine Learning in Highly Automated Driving”. In: *Computer Safety, Reliability, and Security : SAFECOMP 2017 Workshops, ASSURE, DECSoS, SASSUR, TELERISE, and TIPS, Trento, Italy, September 12, 2017, Proceedings*. Ed. by Stefano Tonetta, Erwin Schoitsch, and Friedemann Bitsch. Cham: Springer International Publishing, 2017, pp. 5–16. ISBN: 978-3-319-66284-8. DOI: 10.1007/978-3-319-66284-8_1.

[21] Simon Burton, Lydia Gauerhof, Bibhuti Bhusan Sethy, Ibrahim Habli, and Richard Hawkins. “Confidence Arguments for Evidence of Performance in Machine Learning for Highly Automated Driving Functions”. In: *International Conference on Computer Safety, Reliability, and Security*. Springer. 2019. DOI: 10.1007/978-3-030-26250-1_30.

[22] Simon Burton, Ibrahim Habli, Tom Lawton, John McDermid, Phillip Morgan, and Zoe Porter. “Mind the gaps: Assuring the safety of autonomous systems from an engineering, ethical, and legal perspective”. In: *Artificial Intelligence* 279 (2020), p. 103201. DOI: https://doi.org/10.1016/j.artint.2019.103201.

[23] CARLA. *Open-source simulator for autonomous driving research*. URL: https://carla.org/ (visited on 08/27/2022).

[24] Carmen Carlan, Lydia Gauerhof, Barbara Gallina, and Simon Burton. “Automating Safety Argument Change Impact Analysis for Machine Learning Components”. In: *2022 IEEE 27th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE. 2022, pp. 43–53. DOI: 10.1109/PRDC55274.2022.00019.

[25] Davide Castelvecchi. “Can we open the black box of AI?” In: *Nature* 538.7623 (2016), pp. 20–23. DOI: 10.1038/538020a.

[26] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly detection: A survey”. In: *ACM computing surveys (CSUR)* 41.3 (). ISSN: 0360-0300. DOI: 10.1145/1541880.1541882.

[27] C. Cheng, G. Nhrenberg, C. Huang, H. Ruess, and H. Yasuoka. “Towards Dependability Metrics for Neural Networks”. In: *2018 16th ACM/IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*. Oct. 2018, pp. 1–4. DOI: 10.1109/MEMCOD.2018.8556962.

[28] CNN. *Uber self-driving car operator charged in pedestrian death*. URL: https://edition.cnn.com/2020/09/18/cars/uber-vasquez-charged/index.html (visited on 08/20/2022).

[29] European Commission, Joint Research Centre, S Nativi, and S De Nigris. *AI Watch, AI standardisation landscape state of play and link to the EC proposal for an AI regulatory framework*. Publications Office, 2021. DOI: 10.2760/376602.

[30] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223. DOI: 10.1109/CVPR.2016.350.

[31] Ines Couso, Christian Borgelt, Eyke Hullermeier, and Rudolf Kruse. "Fuzzy sets in data analysis: From statistical foundations to machine learning". In: *IEEE Computational Intelligence Magazine* 14.1 (2019), pp. 31–44. DOI: 10.1109/MCI.2018.2881642.

[32] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. "Underspecification Presents Challenges for Credibility in Modern Machine Learning". In: *arXiv* (2020). DOI: https://doi.org/10.48550/arXiv.2011.03395.

[33] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. "Pedestrian detection: An evaluation of the state of the art". In: *IEEE transactions on pattern analysis and machine intelligence* 34.4 (2011), pp. 743–761. DOI: 10.1109/TPAMI.2011.155.

[34] Katherine Driggs-Campbell and Ruzena Bajcsy. "Communicating intent on the road through human-inspired control schemes". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 3042–3047. DOI: 10.1109/IROS.2016.7759471.

[35] Tom Fawcett. "An introduction to ROC analysis". In: *Pattern recognition letters* 27.8 (2006), pp. 861–874. DOI: 10.1016/j.patrec.2005.10.010.

[36] GM Fitch, SE Lee, S Klauer, J Hankey, J Sudweeks, and T Dingus. *Analysis of lane-change crashes and near-crashes*. 2009. URL: https://www.nhtsa.gov/sites/nhtsa.gov/files/811147.pdf (visited on 10/25/2022).

[37] Fox News. *Tesla smashes into overturned truck while on Autopilot, report says*. URL: https://www.foxnews.com/auto/tesla-smashes-overturned-truck-autopilot (visited on 10/25/2022).

[38] Benoıt Frénay, Ata Kabán, et al. "A comprehensive introduction to label noise." In: *ESANN*. Citeseer. 2014. ISBN: 978-287419095-7.

[39] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. "A review on deep learning techniques applied to semantic segmentation". In: *arXiv* (2017). DOI: 10.48550/arXiv.1704.06857.

[40] Lydia Gauerhof, Roman Gansch, Christian Heinzemann, Matthias Woehrle, and Andreas Heyl. "On the Necessity of Explicit Artifact Links in Safety Assurance Cases for Machine Learning". In: *2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW).* ©2021 IEEE; Reprinted, with permission. 2021, pp. 23–28. DOI: 10.1109/ISSREW53611.2021.00069.

[41] Lydia Gauerhof and Nianlong Gu. *Reverse Variational Autoencoder for Visual Attribute Manipulation and Anomaly Detection.* Winter Application Conference on Computer Vision. ©2020 IEEE; Reprinted, with permission, 2020. DOI: 10.1109/WACV45572.2020.9093319.

[42] Lydia Gauerhof, Yuki Hagiwara, Christoph Schorn, and Mario Trapp. "Considering Reliability of Deep Learning Function to Boost Data Suitability and Anomaly Detection". In: *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW).* ©2020 IEEE; Reprinted, with permission. IEEE. 2020, pp. 249–254. DOI: 10.1109/ISSREW51248.2020.00081.

[43] Lydia Gauerhof, Richard Hawkins, Chiara Picardi, Colin Paterson, Yuki Hagiwara, and Ibrahim Habli. "Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings". In: *Computer Safety, Reliability, and Security.* Reproduced with permission from Springer Nature. Springer International Publishing, 2020. DOI: 10.1007/978-3-030-54549-9_13.

[44] Lydia Gauerhof, Peter Munk, and Simon Burton. "Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving". In: *Computer Safety, Reliability, and Security.* Springer Nature. Springer International Publishing, 2018. DOI: 10.1007/978-3-319-99130-6_4.

[45] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna M. Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. "A Survey of Uncertainty in Deep Neural Networks". In: *CoRR* abs/2107.03342 (2021). DOI: https://doi.org/10.48550/arXiv.2107.03342. arXiv: 2107.03342.

[46] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT press, 2016. DOI: 10.1007/s10710-017-9314-z.

[47] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. "A survey on instance segmentation: state of the art". In: *International journal of multimedia information retrieval* 9.3 (2020), pp. 171–189. DOI: 10.1007/s13735-020-00195-x.

[48] Shijie Hao, Yuan Zhou, and Yanrong Guo. "A brief survey on semantic segmentation with deep learning". In: *Neurocomputing* 406 (2020), pp. 302–321. DOI: 10.1016/j.neucom.2019.11.118.

[49] Richard Hawkins, Tim Kelly, John Knight, and Patrick Graydon. “A new approach to creating clear safety arguments”. In: *Advances in systems safety*. Springer, 2011, pp. 3–23. DOI: 10.1007/978-0-85729-133-2_1.

[50] Dan Hendrycks and Thomas Dietterich. “Benchmarking Neural Network Robustness to Common Corruptions and Surface Variations”. In: *arXiv preprint arXiv:1807.01697* (2018). DOI: https://doi.org/10.48550/arXiv.1807.01697.

[51] Jens Henriksson, Markus Borg, and Cristofer Englund. “Automotive Safety and Machine Learning: Initial Results from a Study on How to Adapt the ISO 26262 Safety Standard”. In: *Proceedings of the 1st International Workshop on Software Engineering for AI in Autonomous Systems*. SEFAIS ’18. Gothenburg, Sweden: Association for Computing Machinery, 2018, pp. 47–49. ISBN: 9781450357395. DOI: 10.1145/3194085.3194090.

[52] Eyke Hüllermeier and Willem Waegeman. “Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods”. In: *Machine Learning* 110.3 (2021), pp. 457–506. DOI: 10.1007/s10994-021-05946-3.

[53] ISO Org. *ISO 21448 Road vehicles — Safety of the intended functionality*. URL: https://www.iso.org/standard/77490.html (visited on 07/24/2022).

[54] ISO Org. *ISO/AWI 8800 Road vehicles — Safety and artificial intelligence*. URL: https://www.iso.org/standard/83303.html (visited on 08/01/2022).

[55] ISO Org. *ISO/IEC 22989:2022 Information technology — Artificial intelligence — Artificial intelligence concepts and terminology*. URL: https://www.iso.org/standard/74296.html (visited on 08/01/2022).

[56] ISO Org. *ISO/IEC 23053:2022 Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)*. URL: https://www.iso.org/standard/74438.html (visited on 08/01/2022).

[57] ISO Org. *ISO/IEC DTR 5469 Artificial intelligence — Functional safety and AI systems*. URL: https://www.iso.org/standard/81283.html (visited on 09/19/2022).

[58] ISO Org. *ISO 26262 Road vehicles - Functional safety*. 2011. URL: https://www.iso.org/standard/68383.html (visited on 10/25/2022).

[59] Ehsan Javanmardi, Yanlei Gu, Mahdi Javanmardi, and Shunsuke Kamijo. “Autonomous vehicle self-localization based on abstract map and multi-channel LiDAR in urban area”. In: *IATSS research* 43.1 (2019), pp. 1–13. DOI: 10.1016/j.iatssr.2018.05.001.

[60] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”. In: *arXiv:1710.10196* (2017). DOI: https://doi.org/10.48550/arXiv.1710.10196.

[61] Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks". In: *Computer Aided Verification*. Ed. by Rupak Majumdar and Viktor Kunčak. Cham: Springer International Publishing, 2017, pp. 97–117. ISBN: 978-3-319-63387-9. DOI: 10.1007/978-3-319-63387-9_5.

[62] Alex Kendall and Yarin Gal. "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" In: *CoRR* abs/1703.04977 (2017). DOI: 10.48550/arXiv.1703.04977.

[63] KI Absicherung. *KI Absicherung*. URL: https://www.ki-absicherung-projekt.de/en/project (visited on 10/25/2022).

[64] Takayuki Kondoh, Tomohiro Yamamura, Satoshi Kitazaki, Nobuyuki Kuge, and Erwin Roeland Boer. "Identification of Visual Cues and Quantification of Drivers' Perception of Proximity Risk to the Lead Vehicle in Car-Following Situations". In: *Journal of Mechanical Systems for Transportation and Logistics* 1.2 (2008), pp. 170–180. DOI: 10.1299/jmtl.1.170.

[65] Philip Koopman and Michael Wagner. "Challenges in autonomous vehicle testing and validation". In: *SAE International Journal of Transportation Safety* 4.1 (2016), pp. 15–24. DOI: 10.4271/2016-01-0128.

[66] Iuliia Kotseruba, Amir Rasouli, and John K. Tsotsos. "Joint Attention in Autonomous Driving (JAAD)". In: *CoRR* abs/1609.04741 (2016). DOI: 10.48550/arXiv.1609.04741. arXiv: 1609.04741.

[67] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. 2009. URL: http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf (visited on 10/25/2022).

[68] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger. Vol. 25. Curran Associates, Inc., 2012. DOI: 10.1145/3065386.

[69] Solomon Kullback and Richard A Leibler. "On Information and Sufficiency". In: vol. 22. 1. The Institute of Mathematical Statistics, Mar. 1951, pp. 79–86. DOI: 10.1214/aoms/1177729694.

[70] Frederic Lambert. *Understanding the fatal Tesla accident on Autopilot and the NHTSA probe*. 2016. URL: https://electrek.co/2016/07/01/understanding-fatal-tesla-accident-autopilot-nhtsa-probe/ (visited on 10/25/2022).

[71] Jean-Claude Laprie. "Dependability: Basic concepts and terminology". In: *Dependability: Basic Concepts and Terminology*. Springer, 1992. DOI: 10.1007/978-3-7091-9170-5_1.

[72] Michael Truong Le, Frederik Diehl, Thomas Brunner, and Alois Knol. "Uncertainty estimation for deep neural object detectors in safety-critical applications". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3873–3878. DOI: 10.1109/ITSC.2018.8569637.

[73] Shixia Liu, Xiting Wang, Mengchen Liu, and Jun Zhu. "Towards better analysis of machine learning models: A visual analytics perspective". In: *Visual Informatics* 1.1 (2017), pp. 48–56. ISSN: 2468-502X. DOI: 10.1016/j.visinf.2017.01.006.

[74] Yang Liu, Peng Sun, Nickolas Wergeles, and Yi Shang. "A survey and performance evaluation of deep learning methods for small object detection". In: *Expert Systems with Applications* 172 (2021), p. 114602. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2021.114602.

[75] Maria Lyssenko, Christoph Gladisch, Christian Heinzemann, Matthias Woehrle, and Rudolph Triebel. "From evaluation to verification: Towards task-oriented relevance metrics for pedestrian detection in safety-critical domains". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 38–45. DOI: 10.1109/CVPRW53098.2021.00013.

[76] Maria Lyssenko, Christoph Gladisch, Christian Heinzemann, Matthias Woehrle, and Rudolph Triebel. "Towards Safety-Aware Pedestrian Detection in Autonomous Systems". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022. DOI: 10.1109/IROS47612.2022.9981309.

[77] G. Marcus. "Deep Learning: A Critical Appraisal". In: *ArXiv e-prints* (2018). DOI: https://doi.org/10.48550/arXiv.1801.00631.

[78] John A McDermid, Yan Jia, Zoe Porter, and Ibrahim Habli. "Artificial intelligence explainability: the technical and ethical dimensions". In: *Philosophical Transactions of the Royal Society A* 379.2207 (2021), p. 20200363. DOI: https://doi.org/10.1098/rsta.2020.0363.

[79] Dimitris Milakis, Bart Van Arem, and Bert Van Wee. "Policy and society related implications of automated driving: A review of literature and directions for future research". In: *Journal of Intelligent Transportation Systems* 21.4 (2017), pp. 324–348. DOI: 10.1080/15472450.2017.1291351.

[80] Ministry of Defence, UK. *Defence Standard 00-56 Safety Management Requirements for Defence Systems. Issue 4*. URL: http://safety.inge.org.uk/20071115-Inge2007a_The_Safety_Case-U.pdf.

[81] Martin Mundt, Yongwon Hong, Iuliia Pliushch, and Visvanathan Ramesh. "A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning". In: *Neural Networks* 160 (2023), pp. 306–336. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2023.01.014.

[82] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. *Reading digits in natural images with unsupervised feature learning*. 2011. URL: http://ai.stanford.edu/~twangcat/papers/nips2011_housenumbers.pdf.

[83] Vu-Linh Nguyen, Mohammad Hossein Shaker, and Eyke Hüllermeier. "How to measure uncertainty in uncertainty sampling for active learning". In: *Machine Learning* 111.1 (2022), pp. 89–122. DOI: 10.1007/s10994-021-06003-9.

[84] Rebecca L Olson, Richard J Hanowski, Jeffrey S Hickman, Joseph Bocanegra, et al. *Driver distraction in commercial vehicle operations*. 2009. URL: https://www.fmcsa.dot.gov/sites/fmcsa.dot.gov/files/docs/FMCSA-RRR-09-042.pdf.

[85] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. *Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. 2019.

[86] Buu Phan, Samin Khan, Rick Salay, and Krzysztof Czarnecki. "Bayesian uncertainty quantification with synthetic data". In: *International Conference on Computer Safety, Reliability, and Security*. Springer. 2019, pp. 378–390. DOI: 10.1007/978-3-030-26250-1_31.

[87] Deborah Prince and Philip Koopman. *Standard for Safety for the Evaluation of Autonomous Products UL4600*. URL: https://edge-case-research.com/ul4600/ (visited on 10/26/2022).

[88] Hilmanda A Putra, Yul Y Nazaruddin, and Endang Juliastuti. "Application of Sensor Fusion for Determining Position and Velocity of Automated People Mover System at Soekarno-Hatta Airport with Extended Kalman Filter". In: *2019 6th International Conference on Instrumentation, Control, and Automation (ICA)*. IEEE. 2019, pp. 172–175. DOI: 10.1109/ICA.2019.8916691.

[89] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. *Do imagenet classifiers generalize to imagenet?* PMLR. 2019. URL: http://proceedings.mlr.press/v97/recht19a/recht19a.pdf (visited on 10/26/2022).

[90] Christian Roesener, Michael Harth, Hendrik Weber, Johanna Josten, and Lutz Eckstein. "Modelling human driver performance for safety assessment of road vehicle automation". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 735–741. DOI: 10.1109/ITSC.2018.8569669.

[91] Rick Salay and Krzysztof Czarnecki. "Improving ML Safety with Partial Specifications". In: *International Conference on Computer Safety, Reliability, and Security*. Springer. 2019, pp. 288–300. DOI: 10.1007/978-3-030-26250-1_23.

[92] Rick Salay, Krzysztof Czarnecki, Ignacio Alvarez, Maria Soledad Elli, Sean Sedwards, and Jack Weast. *PURSS: Towards Perceptual Uncertainty Aware Responsibility Sensitive Safety with ML.* URL: http://ceur-ws.org/Vol-2560/paper34.pdf (visited on 10/26/2022).

[93] Rick Salay, Krzysztof Czarnecki, Hiroshi Kuwajima, Hirotoshi Yasuoka, Toshihiro Nakae, Vahdat Abdelzad, Chengjie Huang, Maximilian Kahn, and Van Duong Nguyen. "The Missing Link: Developing a Safety Case for Perception Components in Automated Driving". In: *WCX SAE World Congress Experience, SAE International in United States* (2022). DOI: 10.4271/2022-01-0818.

[94] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. "An analysis of ISO 26262: Machine learning and safety in automotive software". In: *SAE Technical Papers* 2018 (2018). DOI: 10.4271/2018-01-1075.

[95] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery". In: *International Conference on Information Processing in Medical Imaging.* Springer. 2017, pp. 146–157. DOI: 10.1007/978-3-319-59050-9_12.

[96] Christoph Schorn and Lydia Gauerhof. "FACER: A Universal Framework for Detecting Anomalous Operation of Deep Neural Networks". In: *Proceedings of the IEEE Inteligent Transportation Systems Conference.* ©2020 IEEE; Reprinted, with permission, 2020. DOI: 10.1109/ITSC45102.2020.9294226.

[97] Adrian Schwaiger, Poulami Sinhamahapatra, Jens Gansloser, and Karsten Roscher. *Is Uncertainty Quantification in Deep Learning Sufficient for Out-of-Distribution Detection?* URL: http://ceur-ws.org/Vol-2640/paper_18.pdf (visited on 10/26/2022).

[98] Gesina Schwalbe. "Verification of Size Invariance in DNN Activations Using Concept Embeddings". In: (2021). Ed. by Ilias Maglogiannis, John Macintyre, and Lazaros Iliadis, pp. 374–386. DOI: 10.1007/978-3-030-79150-6_30.

[99] Sergio Segura, Gordon Fraser, Ana B. Sanchez, and Antonio Ruiz-Cortés. "A Survey on Metamorphic Testing". In: *IEEE Transactions on Software Engineering* 42.9 (2016), pp. 805–824. DOI: 10.1109/TSE.2016.2532875.

[100] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization". In: *2017 IEEE International Conference on Computer Vision (ICCV).* 2017, pp. 618–626. DOI: 10.1109/ICCV.2017.74.

[101] Jingyu Shao, Qing Wang, and Fangbing Liu. "Learning to Sample: An Active Learning Framework". In: (2019), pp. 538–547. DOI: 10.1109/ICDM.2019.00064.

[102] Rakshith Shetty, Mario Fritz, and Bernt Schiele. "Towards Automated Testing and Robustification by Semantic Adversarial Data Generation". In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm. Cham: Springer International Publishing, 2020, pp. 489–506. ISBN: 978-3-030-58536-5. DOI: 10.1007/978-3-030-58536-5_29.

[103] Sebastian Söntges and Matthias Althoff. "Computing the Drivable Area of Autonomous Road Vehicles in Dynamic Road Scenes". In: *IEEE Transactions on Intelligent Transportation Systems* 19.6 (2018), pp. 1855–1866. DOI: 10.1109/TITS.2017.2742141.

[104] Bernd Spanfelner, Detlev Richter, Susanne Ebel, Ulf Wilhelm, and Carsten Patz. *Challenges in applying the ISO 26262 for driver assistance systems*. 2012. URL: https://mediatum.ub.tum.de/doc/1142106/1142106.pdf (visited on 10/26/2022).

[105] Niklas Strand, Josef Nilsson, I.C. MariAnne Karlsson, and Lena Nilsson. "Semi-automated versus highly automated driving in critical situations caused by automation failures". In: *Transportation Research Part F: Traffic Psychology and Behaviour* 27 (2014). Vehicle Automation and Driver Behaviour, pp. 218–228. ISSN: 1369-8478. DOI: 10.1016/j.trf.2014.04.005.

[106] Vaibhav Swaminathan, Shrey Arora, Ravi Bansal, and R Rajalakshmi. "Autonomous driving system with road sign recognition using convolutional neural networks". In: *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*. IEEE. 2019, pp. 1–4. DOI: 10.1109/ICCIDS.2019.8862152.

[107] Ömer Şahin Taş, Florian Kuhnt, J. Marius Zöllner, and Christoph Stiller. "Functional system architectures towards fully automated driving". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. 2016, pp. 304–309. DOI: 10.1109/IVS.2016.7535402.

[108] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. SAE. 2021. URL: https://www.sae.org/standards/content/j3016_202104/ (visited on 10/26/2022).

[109] Tesla. *Tesla Deaths*. URL: https://www.tesladeaths.com/ (visited on 10/26/2022).

[110] Stephen Thomas and Katrina M Groth. "Toward a hybrid causal framework for autonomous vehicle safety analysis". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* (2021). DOI: 10.1177/1748006X211043310.

[111] Yonglong Tian, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Pedestrian Detection Aided by Deep Learning Semantic Tasks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015. DOI: 10.1109/CVPR.2015.7299143.

[112] Cumhur Erkan Tuncali, James Kapinski, Hisahiro Ito, and Jyotirmoy V. Deshmukh. “INVITED: Reasoning about Safety of Learning-Enabled Components in Autonomous Cyber-physical Systems”. In: *Design Automation Conference*. 2018. DOI: 10.1109/DAC.2018.8465843.

[113] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. “Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving”. In: *2015 IEEE 18th international conference on intelligent transportation systems*. IEEE. 2015, pp. 982–988. DOI: 10.1109/ITSC.2015.164.

[114] Simon Ulbrich, Tobias Nothdurft, Markus Maurer, and Peter Hecker. “Graph-based context representation, environment modeling and information aggregation for automated driving”. In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE. 2014, pp. 541–547. DOI: 10.1109/IVS.2014.6856556.

[115] Caterina Urban and Antoine Miné. “A Review of Formal Methods applied to Machine Learning”. In: *CoRR* abs/2104.02466 (2021). DOI: 10.48550/arXiv.2104.02466.

[116] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William “Red” Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. “Autonomous Driving in Urban Environments: Boss and the Urban Challenge”. In: (2009). Ed. by Martin Buehler, Karl Iagnemma, and Sanjiv Singh, pp. 1–59. DOI: 10.1007/978-3-642-03991-1_1.

[117] *Verkehrsunfälle - Fachserie 8 Reihe 7 - 2021*. Destatis, Statistisches Bundesamt Deutschland. URL: https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Verkehrsunfaelle/Publikationen/Downloads-Verkehrsunfaelle/verkehrsunfaelle-jahr-2080700217004.pdf?__blob=publicationFile (visited on 10/17/2022).

[118] Hermann Winner, Walther Wachenfeld, and Phillip Junietz. “Validation and introduction of automated driving”. In: *Automotive Systems Engineering II*. Springer, 2018, pp. 177–196. DOI: 10.1007/978-3-319-61607-0_8.

[119] Mirja Wolf, Luiz R. Douat, and Michael Erz. “Safety-Aware Metric for People Detection”. In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. 2021, pp. 2759–2765. DOI: 10.1109/ITSC48978.2021.9564734.

[120] Keisuke Yoneda, Naoki Suganuma, Ryo Yanase, and Mohammad Aldibaja. “Automated driving recognition technologies for adverse weather conditions”. In: *IATSS research* 43.4 (2019), pp. 253–262. DOI: 10.1016/j.iatssr.2019.11.005.

[121] Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. "Adversarially Learned Anomaly Detection". In: (2018), pp. 727–736. DOI: 10.1109/ICDM.2018.00088.

[122] Ran Zhang, Andreas Albrecht, Jonathan Kausch, Henrik J. Putzer, Thomas Geipel, and Prashanth Halady. "DDE process: A requirements engineering approach for machine learning in automated driving". In: *2021 IEEE 29th International Requirements Engineering Conference (RE)*. 2021, pp. 269–279. DOI: 10.1109/RE51729.2021.00031.

[123] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. "CityPersons: A Diverse Dataset for Pedestrian Detection". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4457–4465. DOI: 10.1109/CVPR.2017.474.

[124] Xinhai Zhang, Jianbo Tao, Kaige Tan, Martin Torngren, Jose Manuel Gaspar Sanchez, Muhammad Rusyadi Ramli, Xin Tao, Magnus Gyllenhammar, Franz Wotawa, Naveen Mohan, Mihai Nica, and Hermann Felbinger. "Finding Critical Scenarios for Automated Driving Systems: A Systematic Mapping Study". In: *IEEE Transactions on Software Engineering* (2022). DOI: 10.1109/TSE.2022.3170122.

[125] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. "Object Detection With Deep Learning: A Review". In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), pp. 3212–3232. DOI: 10.1109/TNNLS.2018.2876865.

A Credits

Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving

This article (Section B.1) is authored by Lydia Gauerhof, Peter Munk and Simon Burton and published by Springer in the proceedings of SafeComp 2018 [44].

Lydia Gauerhof contributed about 80 % of the article's content. She developed the conception, selection and description of gaps and corresponding validation targets and the conception of the evidence.

Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings

This article (Section B.2) is authored by Lydia Gauerhof, Richard Hawkins, Chiara Picardi, Colin Paterson, Yuki Hagiwara and Ibrahim Habli and published by Springer in the proceedings of SafeComp 2020 [43]. Lydia Gauerhof contributed about 35 % of the article's content. She developed the use case, the selected scenario, the concritization of problem statement and she advanced requirements elicitation. She led and sharpened the discussion on how the requirements drive the safety activities in the ML-lifecycle. She developed the concept and conducted the experiments with the help of Yuki Hagiwara.

Considering Reliability of Deep Learning Function to Boost Data Suitability and Anomaly Detection

This article (Section B.3) is authored by Lydia Gauerhof, Yuki Hagiwara, Christoph Schorn and Mario Trapp and published by IEEE in the proceedings of ISSREW 2020 [42]. Lydia Gauerhof contributed about 60 % of the article's content. She developed the conception and definition of data suitability, systematic errors and corresponding counter measure, the concept of experiments. She conducted the experiments with the support of Yuki Hagiwara.

FACER: A Universal Framework for Detecting Anomalous Operation of Deep Neural Networks

This article (Section B.4) is authored by Lydia Gauerhof and Christoph Schorn. The article is published by IEEE in the proceedings of ITSC 2020 [96]. Lydia Gauerhof contributed about 50 % of the article's content. She initialized the idea of feature based anomaly detection on corrupted input data and contributed the final conception and development of FACER together with Christoph Schorn.

Reverse Variational Autoencoder for Visual Attribute Manipulation and Anomaly Detection

This article (Section B.5) is authored by Lydia Gauerhof and Nianlong Gu. The article is published by IEEE in the proceedings of WACV 2020 [41]. Lydia Gauerhof contributed about 50 % of the article's content. She initialized to main concept and contributed to the development and description of ReverseVAE together with Nianlong Gu.

Testing Deep Learning-based Visual Perception for Automated Driving

This article (Section B.6) is authored by Stephanie Abrecht, Lydia Gauerhof, Christoph Gladisch, Konrad Groh, Christian Heinzemann, Matthias Woehrle. The article is published by ACM in the journal of ACM Transactions on Embedded Computing Systems 2021 [2]. Lydia Gauerhof is a co-author to this article and contributed about 15 % of the article's content. She mainly shaped the section on test oracle based on her expertise. Furthermore, she contributed to other sections, such as test input generation and coverage.

On the Necessity of Explicit Artifact Links in Safety Assurance Cases for Machine Learning

This article (Section B.7) is authored by Lydia Gauerhof, Romand Gansch, Christian Heinzemann, Matthias Woehrle, Andreas Heyl. The article is published by IEEE in the proceedings of ISSREW 2021 [40]. Lydia Gauerhof contributed about 60 % of the article's content. She elaborated the conception of explicit artefact links and she strongly drove the examples and the discussion of the evaluation.

B Original Manuscripts

This appendix provides all publications that are presented in Chapters 4 and 5. Publications are arranged according to their appearance in this thesis.

Article [44] in Section B.1 and article [43] in Section B.2 are reproduced with permission from Springer Nature.

Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving

Lydia Gauerhof, Peter Munk, and Simon Burton

Corporate Research Robert Bosch GmbH
Robert-Bosch-Campus 1, 71272 Renningen, Germany
{firstname.lastname}@de.bosch.com

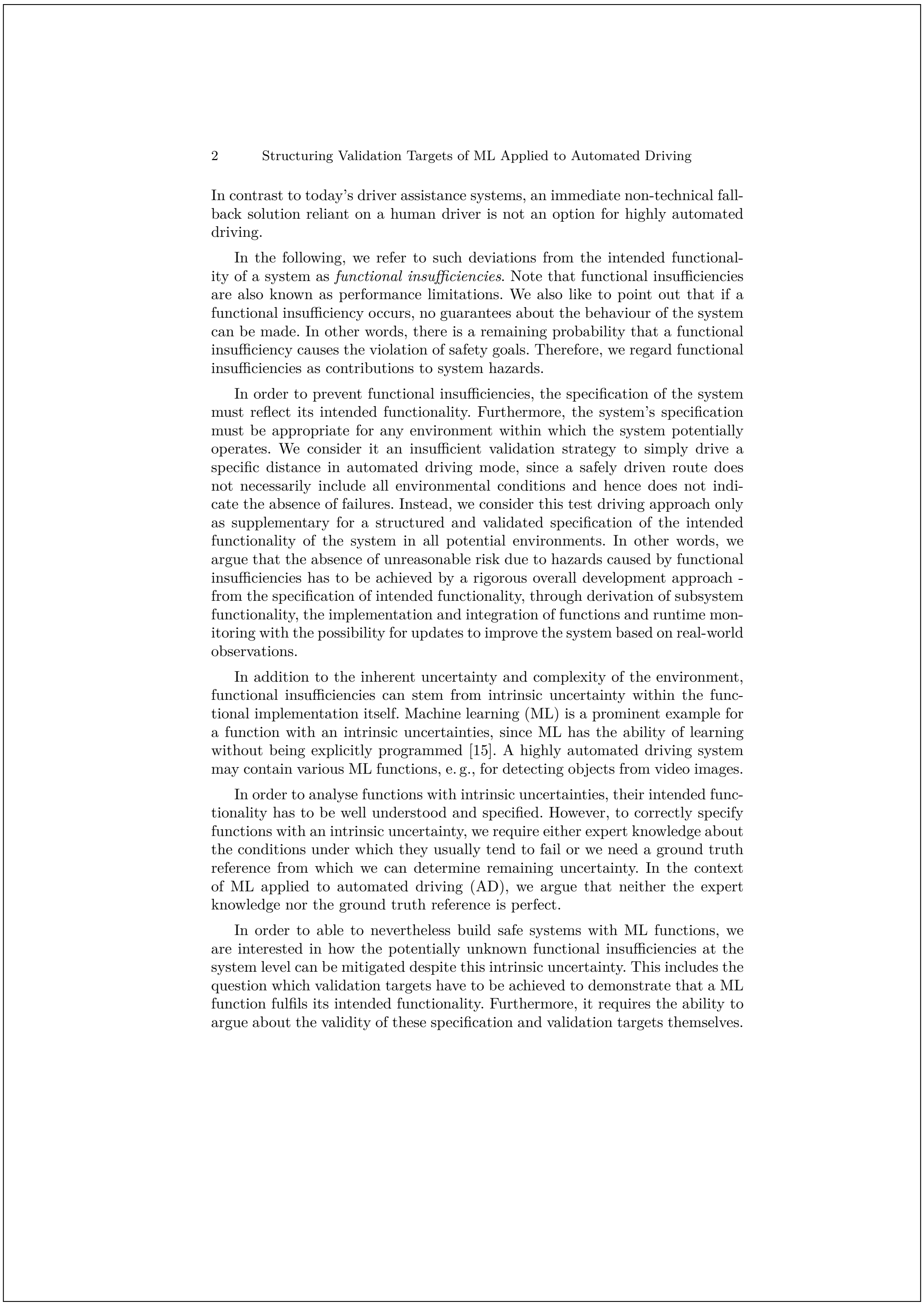
Abstract. The validation of highly automated driving vehicles is an important challenge to the automotive industry, since even if the system is free from internal faults, its behaviour might still vary from the original intent. Reasons for these deviations from the intended functionality can be found in the unpredictabiltiy of environmental conditions as well the intrinsic uncertainties of the Machine Learning (ML) functions used to make sense of this complex input space.
In this paper, we propose a safety assurance case for a pedestrian detection function, a safety-relevant baseline functionality for an automated driving system. Our safety assurance case is presented in the graphical structuring notation (GSN) and combines our arguments against the problems of underspecification [9], the semantic gap [3], and the deductive gap [16].

Keywords: safety, intended functionality, functional insufficiency, nominal performance, automated driving, Machine Learning, assuance case, GSN

1 Introduction

Highly automated driving vehicles will potentially ease the daily life of millions of commuters, increase the mobility of elderly and disabled people, and enable numerous new business cases. A highly automated driving system can be defined as a vehicle that monitors its driving environment and executes steering, acceleration and deceleration without permanent human monitoring or intervention.

In order to achieve the safety goals of a highly automated driving system, e. g., do not harm pedestrians, the propagation of internal faults must be prevented as is standard in today's automotive systems. However, we must additionally deal with situations where the automated driving system is free from internal faults but behaves in a manner that nevertheless leads to a hazard. For example, a cold and foggy environment can result in blurred camera images and the radar sensors becoming iced such that a safe automated driving operation cannot be ensured and the probability of the system violating its safety goals is unacceptably high.



In contrast to today's driver assistance systems, an immediate non-technical fall-back solution reliant on a human driver is not an option for highly automated driving.

In the following, we refer to such deviations from the intended functionality of a system as *functional insufficiencies*. Note that functional insufficiencies are also known as performance limitations. We also like to point out that if a functional insufficiency occurs, no guarantees about the behaviour of the system can be made. In other words, there is a remaining probability that a functional insufficiency causes the violation of safety goals. Therefore, we regard functional insufficiencies as contributions to system hazards.

In order to prevent functional insufficiencies, the specification of the system must reflect its intended functionality. Furthermore, the system's specification must be appropriate for any environment within which the system potentially operates. We consider it an insufficient validation strategy to simply drive a specific distance in automated driving mode, since a safely driven route does not necessarily include all environmental conditions and hence does not indicate the absence of failures. Instead, we consider this test driving approach only as supplementary for a structured and validated specification of the intended functionality of the system in all potential environments. In other words, we argue that the absence of unreasonable risk due to hazards caused by functional insufficiencies has to be achieved by a rigorous overall development approach - from the specification of intended functionality, through derivation of subsystem functionality, the implementation and integration of functions and runtime monitoring with the possibility for updates to improve the system based on real-world observations.

In addition to the inherent uncertainty and complexity of the environment, functional insufficiencies can stem from intrinsic uncertainty within the functional implementation itself. Machine learning (ML) is a prominent example for a function with an intrinsic uncertainties, since ML has the ability of learning without being explicitly programmed [15]. A highly automated driving system may contain various ML functions, e. g., for detecting objects from video images.

In order to analyse functions with intrinsic uncertainties, their intended functionality has to be well understood and specified. However, to correctly specify functions with an intrinsic uncertainty, we require either expert knowledge about the conditions under which they usually tend to fail or we need a ground truth reference from which we can determine remaining uncertainty. In the context of ML applied to automated driving (AD), we argue that neither the expert knowledge nor the ground truth reference is perfect.

In order to able to nevertheless build safe systems with ML functions, we are interested in how the potentially unknown functional insufficiencies at the system level can be mitigated despite this intrinsic uncertainty. This includes the question which validation targets have to be achieved to demonstrate that a ML function fulfils its intended functionality. Furthermore, it requires the ability to argue about the validity of these specification and validation targets themselves.

Different challenges in AD development have been already described: underspecification [9], semantic gap [3] and deductive gap [16]. In this paper we review these contributions by means of an ML-specific case study for the safety relevant function “pedestrian detection”. Based on this case study, we propose approaches to answer the following questions: (i) Underspecification: What is the intended functionality and what are its limits? (ii) Semantic gap: How can the intended functionality be described? (iii) Deductive gap: Which requirements on the functional layer (here: ML) can be deduced?

We present a safety assurance case that supports the argument of the absence of unreasonable risk due to hazards caused by functional insufficiencies by structuring the validation targets. Our safety assurance case broadens the approach of Burton et al. [4] and is visualised by a graphical notation, namely the goals structuring notation (GSN).

The remainder of this paper is structured as follows: In Section 2, we detail the function pedestrian detection of our case study and derive validation targets for it. In Section 3, we present the safety assurance case of the pedestrian detection function. In Section 4, we summarize our results, discuss our approach, and present future work.

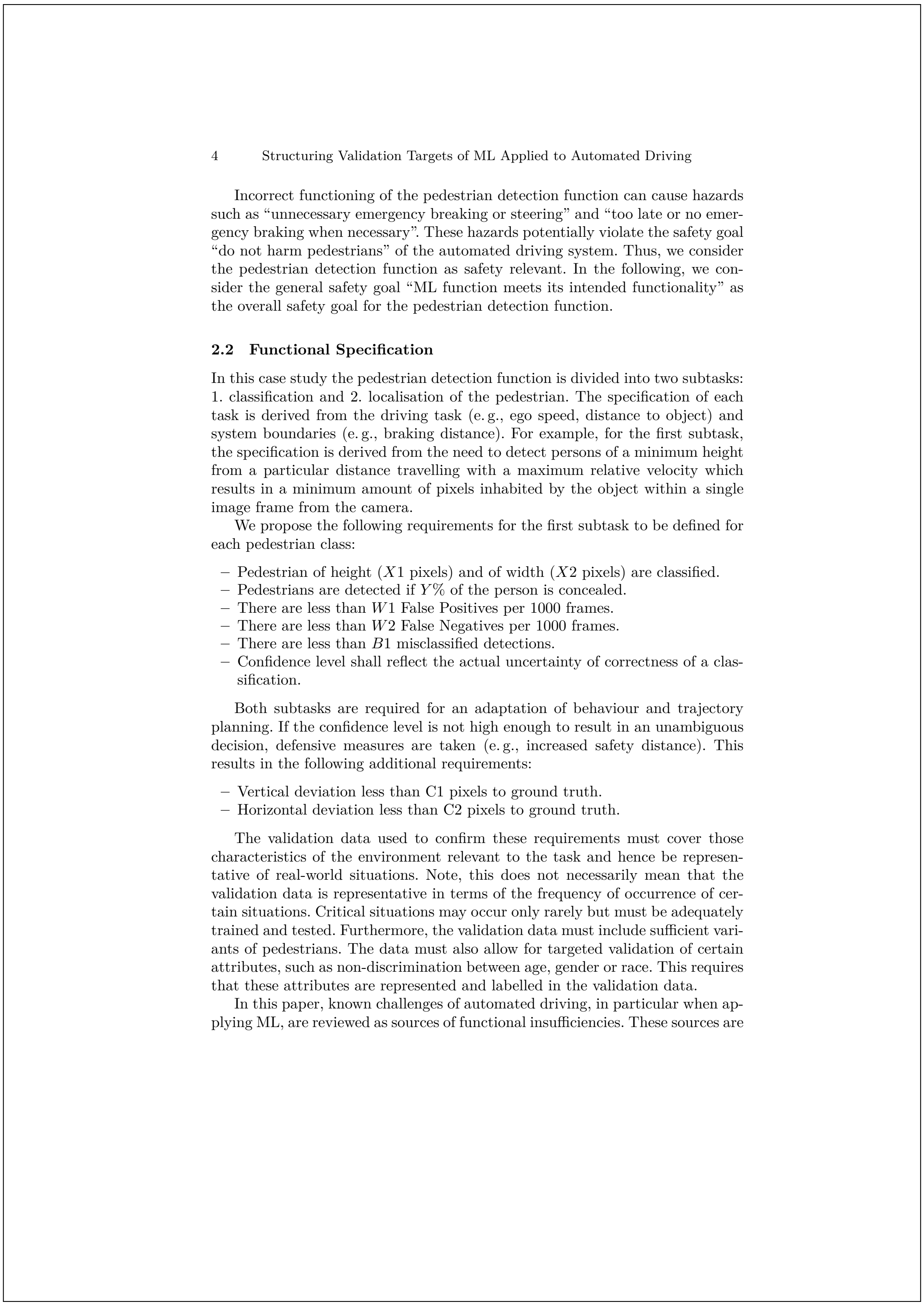
2 Validation Targets for Pedestrian Detection

In this section, we first introduce the pedestrian detection function and present its functional specification. Then we discuss reducing the risk of hazards caused by underspecification and semantic gap. Finally, we discuss the deductive gap and propose functional modifications to achieve the intended functionality.

2.1 The Pedestrian Detecting Function

A typical automated driving system is comprised of the following parts: sensors, perception, behaviour and trajectory planning, trajectory control, and actuators. The perception part acquires and processes data from sensors, e. g., cameras, lidars, and radar sensors, and other data sources, e. g., car2X-communication, and creates an environmental model of the surroundings of the vehicle. For our case study, we focus on the ML function that detects pedestrians based on video analysis.

This pedestrian detection function is typically realized by a Convolutional Neural Network (CNN), since CNNs are regarded to have a high potential for classification tasks [10]. CNNs are a class of feed-forward neural networks (NN) that consist of a large number of connected neurons - computational units that calculate a weighted sum of their inputs and apply a nonlinear activation function on this sum. The weights are determined by minimizing a loss function of the network over a given set of training data (labelled images) and backpropagating the respective error terms through the network. In this manner, CNNs allow a classification annotated with a confidence level for each class and a localisation of an object within a given image (e. g., frames of a video).



Incorrect functioning of the pedestrian detection function can cause hazards such as “unnecessary emergency breaking or steering” and “too late or no emergency braking when necessary”. These hazards potentially violate the safety goal “do not harm pedestrians” of the automated driving system. Thus, we consider the pedestrian detection function as safety relevant. In the following, we consider the general safety goal “ML function meets its intended functionality” as the overall safety goal for the pedestrian detection function.

2.2 Functional Specification

In this case study the pedestrian detection function is divided into two subtasks: 1. classification and 2. localisation of the pedestrian. The specification of each task is derived from the driving task (e. g., ego speed, distance to object) and system boundaries (e. g., braking distance). For example, for the first subtask, the specification is derived from the need to detect persons of a minimum height from a particular distance travelling with a maximum relative velocity which results in a minimum amount of pixels inhabited by the object within a single image frame from the camera.

We propose the following requirements for the first subtask to be defined for each pedestrian class:

- Pedestrian of height ($X1$ pixels) and of width ($X2$ pixels) are classified.
- Pedestrians are detected if $Y\%$ of the person is concealed.
- There are less than $W1$ False Positives per 1000 frames.
- There are less than $W2$ False Negatives per 1000 frames.
- There are less than $B1$ misclassified detections.
- Confidence level shall reflect the actual uncertainty of correctness of a classification.

Both subtasks are required for an adaptation of behaviour and trajectory planning. If the confidence level is not high enough to result in an unambiguous decision, defensive measures are taken (e. g., increased safety distance). This results in the following additional requirements:

- Vertical deviation less than C1 pixels to ground truth.
- Horizontal deviation less than C2 pixels to ground truth.

The validation data used to confirm these requirements must cover those characteristics of the environment relevant to the task and hence be representative of real-world situations. Note, this does not necessarily mean that the validation data is representative in terms of the frequency of occurrence of certain situations. Critical situations may occur only rarely but must be adequately trained and tested. Furthermore, the validation data must include sufficient variants of pedestrians. The data must also allow for targeted validation of certain attributes, such as non-discrimination between age, gender or race. This requires that these attributes are represented and labelled in the validation data.

In this paper, known challenges of automated driving, in particular when applying ML, are reviewed as sources of functional insufficiencies. These sources are structured into validation targets, so that a systematic approach is introduced to arguing the goal “ML functions meets its intended functionality”. A cumulative argumentation based on a diverse set of validation targets and statistical evaluation (here formulated as sub-goals of a safety assurance case) including the associated evidence must be discussed. In the following, a thorough investigation is made into how to analyse the intended functionality and how to set the validation targets. This approach is later used to build a safety assurance case.

First of all, the risk due to hazards caused by two sources of functional insufficiencies shall be reduced: underspecification and semantic gap. Later the implementation specific deductive gap will be reviewed. These validation targets must be determined iteratively during development.

2.3 Reduction of Risk due to Hazards Caused by Underspecification

Underspecification might occur if the intended functionality is more diverse than what is specified [9]. Consequently, defined use cases are only part of the intended functionality. Addressing underspecification by means of a generalization can lead to a inadequately defined safety requirements. In order to reduce risk of underspecification, we suggest the following validation targets. Note that this list might need to be extended and evaluated for each specific task.

– Environment is sufficiently well known.
  ⇒ Evidence: The hazard analysis and risk assessment (HARA) in the scope of ISO 26262 should be extended to determine properties of the environment that lead to critical situations, e. g., low-angled sunlight, fog and reflective surfaces, etc. Systems safety approaches such STAMP [11] [12]) can be beneficial here. While specifying the intended functionality, unintended use cases must be excluded explicitly in order to highlight the system boundaries. Assumptions on the environment shall be made explicit in order that they can be validated through review, analysis and monitoring.
– Task is sufficiently well known.
  ⇒ Evidence: Requirements shall be specified including task specific attributes. In the case of generalization abilities, attributes such as colour invariances and translations invariances might be required.
– Sensitivity against unpredictable or unspecified impact of environmental attributes is sufficiently low.
  ⇒ Evidence: Sensitivity to environmental changes shall be investigated. Moreover, influence due to distributional shift over time or due to geographic changes shall be reviewed. Requirements on invariance and generalization attributes shall be reviewed according their appropriateness to the intended functionality. Run-time monitoring of assumptions and field-based validation shall be used to investigate discrepancies between the real environment and the assumptions as well as sensitivity to these changes. Moreover, statistical extrapolation shall be used generalise the results of acquired data.

2.4 Reduction of Risk due to Hazards Caused by Semantic Gap

Semantic gap refers to using implicit knowledge on the satisfaction of Safety Goals [3]. In the context of ML, semantic gap refers to making claims on the relevance of references used for training, test and validation data sets. We propose the following sub-goals to support the argument of "Reduction of risk due to semantic gap". Note that these might have to be extended and evaluated:

– Pedestrian classes are sufficiently accurately described.
  ⇒ Evidence: Functional specification of several validation data sub-sets shall include all variants of classes that can be derived from the environment. Moreover, safety requirements shall be transferred into task specific requirements, e.g. informal textual specifications shall be transferred into formal specifications as far as it is possible, at least for safety-relevant requirements. Evaluation of specific influences and appropriate object variations shall be specified beyond statistical evaluation.
– Location accuracy is sufficiently well described.
  ⇒ Evidence: training and validation data shall be specified. Evaluation of specific influences shall be specified. Additionally, evaluation of compliance with tolerances, of size and of location variation shall be specified.
– Discrepancy between real and described environment is sufficiently small.
  ⇒ Evidence: Evaluation of similarity between reality and specification of validation data shall be specified. Functional modifications, such as runtime monitoring, degradation modes, pre-processing of ML input etc., shall be specified and documented.

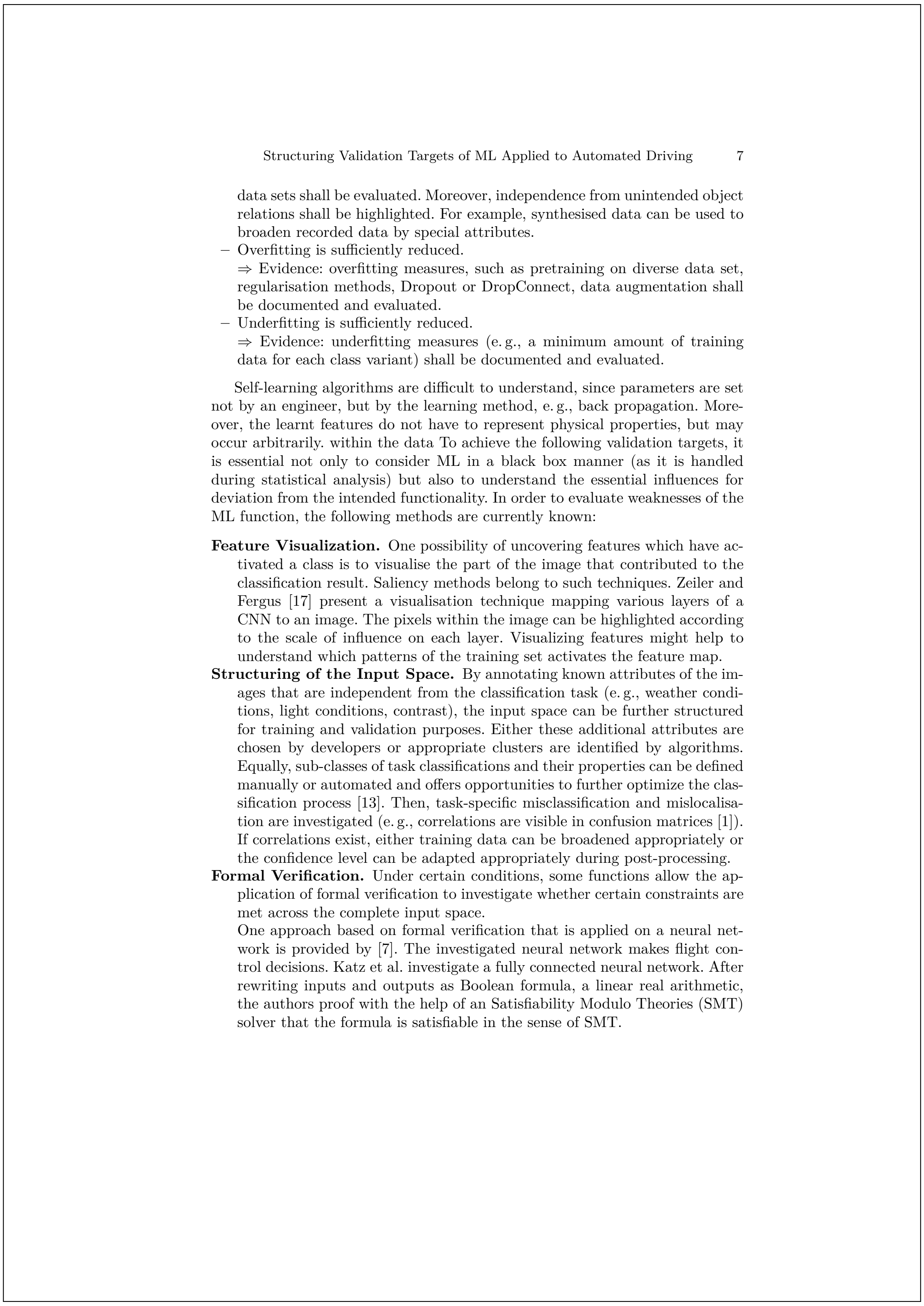
Although systems engineering approaches to ensure a rigorous and complete derivation of the requirements reduce the risk due to hazards caused by underspecification and semantic gap, further evidence must be collected through targeted field-based observations and used to iteratively improve the specifications.

2.5 Reduction of Risk due to Hazards Caused by Deductive Gap

Deductive gap refers to using invalid assumptions on different abstraction levels causing an unintended functionality [16]. In the context of ML, features might be wrongly learnt or erroneously implemented. The deductive gap for ML differs from the deductive gap for non ML functions due to its intrinsic uncertainty. Note that reduction of underspecification and semantic gap is a prerequisite for the implementation of the intended function and as important as the avoidance of deductive gap.

The following validation targets can be defined for reducing the risk due to hazards caused by deductive gap before and during training:

– Data set is sufficient for the intended functionality.
  ⇒ Evidence: Transfer of system-level requirements to ML-specific requirements as well as the attribute distribution within training, test and validation



data sets shall be evaluated. Moreover, independence from unintended object relations shall be highlighted. For example, synthesised data can be used to broaden recorded data by special attributes.

– Overfitting is sufficiently reduced.
  ⇒ Evidence: overfitting measures, such as pretraining on diverse data set, regularisation methods, Dropout or DropConnect, data augmentation shall be documented and evaluated.
– Underfitting is sufficiently reduced.
  ⇒ Evidence: underfitting measures (e. g., a minimum amount of training data for each class variant) shall be documented and evaluated.

Self-learning algorithms are difficult to understand, since parameters are set not by an engineer, but by the learning method, e. g., back propagation. Moreover, the learnt features do not have to represent physical properties, but may occur arbitrarily. within the data To achieve the following validation targets, it is essential not only to consider ML in a black box manner (as it is handled during statistical analysis) but also to understand the essential influences for deviation from the intended functionality. In order to evaluate weaknesses of the ML function, the following methods are currently known:

Feature Visualization. One possibility of uncovering features which have activated a class is to visualise the part of the image that contributed to the classification result. Saliency methods belong to such techniques. Zeiler and Fergus [17] present a visualisation technique mapping various layers of a CNN to an image. The pixels within the image can be highlighted according to the scale of influence on each layer. Visualizing features might help to understand which patterns of the training set activates the feature map.

Structuring of the Input Space. By annotating known attributes of the images that are independent from the classification task (e. g., weather conditions, light conditions, contrast), the input space can be further structured for training and validation purposes. Either these additional attributes are chosen by developers or appropriate clusters are identified by algorithms. Equally, sub-classes of task classifications and their properties can be defined manually or automated and offers opportunities to further optimize the classification process [13]. Then, task-specific misclassification and mislocalisation are investigated (e. g., correlations are visible in confusion matrices [1]). If correlations exist, either training data can be broadened appropriately or the confidence level can be adapted appropriately during post-processing.

Formal Verification. Under certain conditions, some functions allow the application of formal verification to investigate whether certain constraints are met across the complete input space.
One approach based on formal verification that is applied on a neural network is provided by [7]. The investigated neural network makes flight control decisions. Katz et al. investigate a fully connected neural network. After rewriting inputs and outputs as Boolean formula, a linear real arithmetic, the authors proof with the help of an Satisfiability Modulo Theories (SMT) solver that the formula is satisfiable in the sense of SMT.

Nevertheless, in our case study, a formal verification of CNNs is not realistically feasible, since it is difficult to describe the input and output space (images with all kinds of variations in appearance etc.) or to formulate linear real arithmetic for this purpose.

Uncertainty. The confidence level of each class output does not express a probability of existence of the object itself. Therefore, uncertainty calculation might be used to measure the reliability of the classification result. Uncertainty quantification can be used for further measures (e. g., in plausibility checks and sensor fusion algorithms [8]), thus improving the overall robustness and reliability of the subsequent trajectory planning tasks [14].

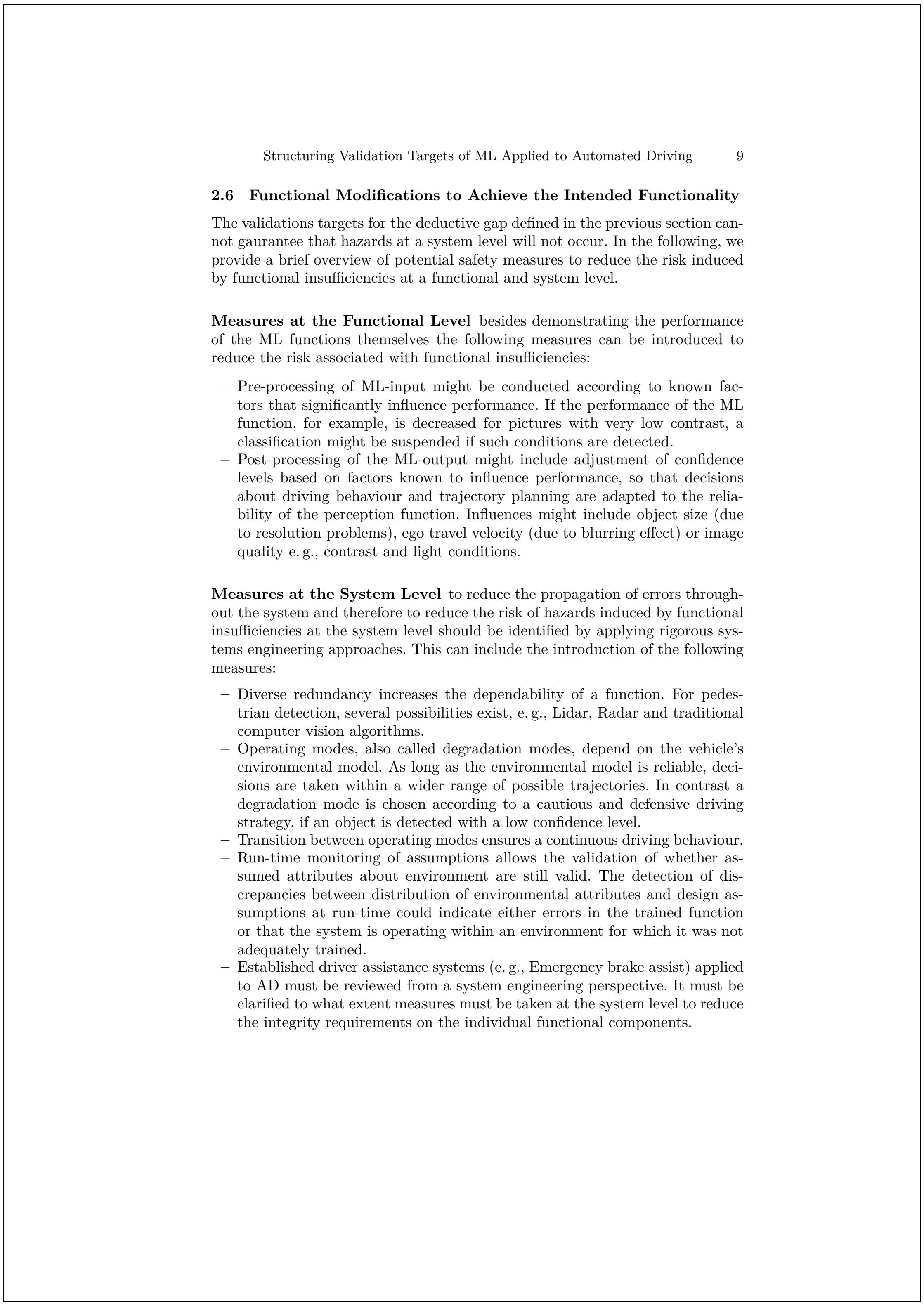
With the help of these methods, the following validation targets must be proven in order to reduce the risk of hazards caused by deductive gap. We suggest the following incomplete list of evidences:

- Essential influences on the ML function are sufficiently understood.
  ⇒ Evidence: The application of feature visualization, adaptation of confidence level and uncertainty calculation shall be documented. Furthermore, correlations of errors to features shall be investigated and reduced by appropriate training. Evaluation of these correlations shall be documented.
- ML function is sufficiently robust.
  ⇒ Evidence: tolerance against distributional shift, adversarial and faulty input shall be evaluated. Statistical evaluation shall be documented. An integrity test of ML function shall be documented.
- Learnt features are sufficient for function.
  ⇒ Evidence: learnt features and correlations between these and detection results shall be analysed and documented (e. g., by feature visualization).

A well-known weakness of ML functions is the sensitivity to adversarial attacks. In this case an object, e. g., a road sign [6] [5], is slightly modified such that a human would not recognize a manipulation but it is misclassified with a high confidence by a machine learning function. ML functions trained with different data subsets or with adversarial examples are more robust against adversarial attacks but unfortunately, to the best of our knowledge, there is no general solution to avoid adversarial attacks as of today. Hence, special caution is necessary while system engineering and when creating validation targets for adversarial attacks.

Furthermore, the following validations targets regarding any changes made during the development of the system (e. g., in its parameters or in the computing platform) must be proven.

- Changes to parameters do not inviolate safety requirements.
  ⇒ Evidence: verification specification for any changes shall be documented.
- Differences between the training and target platforms do not lead to a violation of the safety requirements.
  ⇒ Evidence: verification specification for any changes shall be documented.
- Changes in target platform comply with safety requirements.
  ⇒ Evidence: verification specification for any changes in target platform shall be documented.



2.6 Functional Modifications to Achieve the Intended Functionality

The validations targets for the deductive gap defined in the previous section cannot gaurantee that hazards at a system level will not occur. In the following, we provide a brief overview of potential safety measures to reduce the risk induced by functional insufficiencies at a functional and system level.

Measures at the Functional Level besides demonstrating the performance of the ML functions themselves the following measures can be introduced to reduce the risk associated with functional insufficiencies:

– Pre-processing of ML-input might be conducted according to known factors that significantly influence performance. If the performance of the ML function, for example, is decreased for pictures with very low contrast, a classification might be suspended if such conditions are detected.
– Post-processing of the ML-output might include adjustment of confidence levels based on factors known to influence performance, so that decisions about driving behaviour and trajectory planning are adapted to the reliability of the perception function. Influences might include object size (due to resolution problems), ego travel velocity (due to blurring effect) or image quality e. g., contrast and light conditions.

Measures at the System Level to reduce the propagation of errors throughout the system and therefore to reduce the risk of hazards induced by functional insufficiencies at the system level should be identified by applying rigorous systems engineering approaches. This can include the introduction of the following measures:

– Diverse redundancy increases the dependability of a function. For pedestrian detection, several possibilities exist, e. g., Lidar, Radar and traditional computer vision algorithms.
– Operating modes, also called degradation modes, depend on the vehicle's environmental model. As long as the environmental model is reliable, decisions are taken within a wider range of possible trajectories. In contrast a degradation mode is chosen according to a cautious and defensive driving strategy, if an object is detected with a low confidence level.
– Transition between operating modes ensures a continuous driving behaviour.
– Run-time monitoring of assumptions allows the validation of whether assumed attributes about environment are still valid. The detection of discrepancies between distribution of environmental attributes and design assumptions at run-time could indicate either errors in the trained function or that the system is operating within an environment for which it was not adequately trained.
– Established driver assistance systems (e. g., Emergency brake assist) applied to AD must be reviewed from a system engineering perspective. It must be clarified to what extent measures must be taken at the system level to reduce the integrity requirements on the individual functional components.

3 Safety Assurance Case for Pedestrian Detection

The previous section introduced validation targets that must be addressed during development. Nevertheless, the argumentation that a ML function meets its intended functionality has to be summarised and structured. In this section we propose a safety assurance case using the goal structuring notation (GSN) that structures the presented validation targets and associated evidence. The aim of our safety assurance case is to argue the safety of ML with respect to the absence of unreasonable risk due to hazards caused by functional insufficiencies.

The safety assurance case includes all validation targets as sub-goals and evidence of mitigating against weaknesses in the ML function at the system as well as functional level. An evaluation of the validation targets is not conducted in this paper, since it is not representative for a general validation approach due to environmental-, task-, and system-specifc dependencies. Instead we propose and discuss an approach to structure validation targets for a safety relevant ML function applied to automated driving and to combine diverse sources of evidence.

Figure 1 graphically represents our approach to arguing that the ML function meets its intended functionality. In order to support the goal "Machine Learning function meets all of its functional requirements", the strategy "Argument over sufficient reduction of root causes of functional insufficiencies" is given. This argument is associated with the three sub-goals to reduce risk due to underspecification, semantic gap and deductive gap. Their sub-goals in turn are not depicted, but stated in the following.

In the right upper context symbol all information about the ML function, its tasks, requirements and the CNN are stated (usually as a link to an appropriate document). The use of a CNN for the task is argued by the statement that CNNs are currently the most successful classifier (right lower context symbol) and this statement is justified by the contests that are won by CNNs [10]. Therefore, this justification also depends on the assumption that classification performance from benchmark contests are transferable and therefore highlight the potential of CNNs for classification in general.

The goal "ML function meets its intended functionality" is stated within the context "Pedestrian detection is a safety-relevant function, so that this ML function is also safety-relevant." which might be linked to a hazard and risk analysis. Furthermore, the assumptions on the environment are also stated (usually linked to an appropriate document).

The strategy to achieve the main goal "ML function meets its intended functionality" is argued by a sufficient reduction of root causes of functional insufficiencies. This argument is reflected in the three sub-goals reducing risk due to underspecification, semantic gap and deductive gap with the arguments over appropriate specification, description, deduction. Note that the justification that these three sub-goals are sufficient is missing here, but shall be stated in general.

In figure 2, the GSN is refined for the two arguments over appropriate specification and description. The sub-goal "Reduction of risk due to semantic gap" is achievable for the given specification that reflects the sub-goal "Reduction of risk

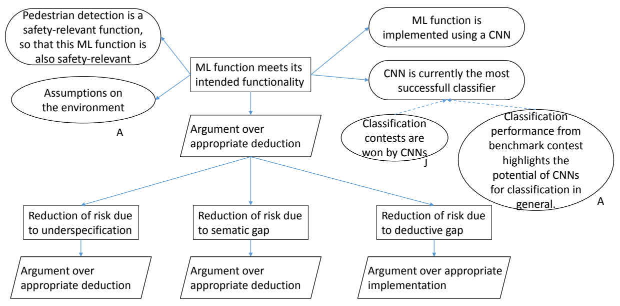

Pedestrian detection is a safety-relevant function, so that this ML function is also safety-relevant
Assumptions on the environment
A
ML function meets its intended functionality
ML function is implemented using a CNN
CNN is currently the most successfull classifier
Classification contests are won by CNNs
J
Classification performance from benchmark contest highlights the potential of CNNs for classification in general.
A
Argument over appropriate deduction
Reduction of risk due to underspecification
Reduction of risk due to sematic gap
Reduction of risk due to deductive gap
Argument over appropriate deduction
Argument over appropriate deduction
Argument over appropriate implementation

Fig. 1. GSN for the Goal Machine learning function meets its intended functionality.

due to underspecification". Moreover, all validation targets reducing the risk associated with underspecification and semantic gap lead to a refined specification that is task-specific, but not implementation-specific.

Figure 3 depicts the sub-goal "Reduction of risk due to deductive gap" focusing on the implemented function. Its refined specification is stated in the context element. The specification holds, since it is assumed that intended functionality is well understood and described. A justification is missing here, but shall be stated in general.

Especially the first validation target "Data set is sufficient for intended function" of the argument over appropriate implementation points out that the data set (that is the basis for training, test and validation data set) shall reflect the goals "Reduction of risk due to underspecification" and "Reduction of risk due to semantic gap". Here the dependencies between all three goals become clear. If underspecification or a semantic gap increases the risk of unintended functionality, then this will also affect the data set. For example, if a firefighter was not explicitly included in the specification, the data set might also lack this kind of variant of a person. Consequently, a ML function is not trained properly and might not be able to recognize the firefighter.

Additionally, the refined specification must also approach the integrity requirements on the individual functions. The awareness of attributes that might be adverse to the intended functionality under particular conditions, might be used to introduce a data fusion with further information processing methods and/or redundant information sources (e. g., sensors, digital maps, server). Therefore, the deductive gap might be mitigated by other measures than within the function itself. These measures are stated in the context element on the left hand side (usually linked to an appropriate document).

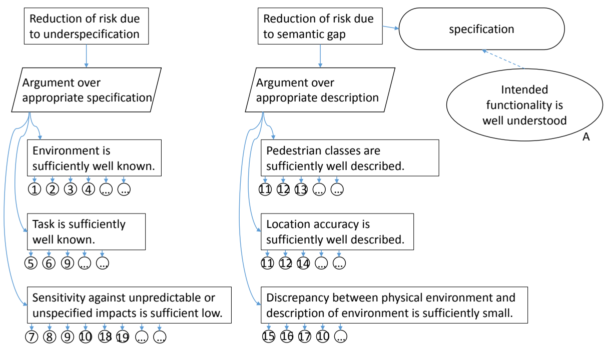

Reduction of risk due to underspecification
Argument over appropriate specification
Environment is sufficiently well known.
1 2 3 4 ... ...
Task is sufficiently well known.
5 6 9 ... ...
Sensitivity against unpredictable or unspecified impacts is sufficient low.
7 8 9 10 18 19 ... ...
Reduction of risk due to semantic gap
specification
Argument over appropriate description
Intended functionality is well understood
A
Pedestrian classes are sufficiently well described.
11 12 13 ... ...
Location accuracy is sufficiently well described.
11 12 14 ... ...
Discrepancy between physical environment and description of environment is sufficiently small.
15 16 17 10 ...

Fig. 2. GSN for the Goal: ”Reduction of risk due to underspecification and semantic gap”. The following evidence is suggested to support the claims: 1) hazard and risk analysis, 2) accident database, 3) explicit exclusion of unintended use cases, 4) explicit assumptions on environment, 5) requirements specification (colour invariances, translations invariances), 6) documentation of functional modifications, 7) sensitivity investigation, 8) review of distributional shift, 9) specification of invariance and generalization attributes, 10) run-time monitoring, 11) specification of training and validation data, 12) evaluation of specific influences, 13) evaluation of appropriate object variations, 14) evaluation of compliance with tolerances, of size and location variation, 15) evaluation of similarity between reality and specification of validation data, 16) degradation modes, 17) pre-processing of ML input, 18) field-based validation, 19) statistical extrapolation

4 Conclusion and Future Work

We investigated sources of functional insufficiencies and derived validation targets in order to demonstrate the intended functionality of a machine learning (ML) function in an automated driving system, namely the pedestrian detection function. Our approach is based on the reduction of the well-known root causes of functional insufficiencies: underspecification [9], the semantic gap [3], and the deductive gap [16]. From the validation targets we outlined a safety assurance case in GSN for the safety goal “Machine learning function meets its intended functionality” of the pedestrian detection function.

When we derived and structured the validation targets for our case study we included well-known methods and measures. While statistical evaluation offers a good basis to investigate improvements at a functional and system level, other methods, such as feature visualization, are used for analysis and for reaching a better overall understanding of the learnt functionality. However, the effective-

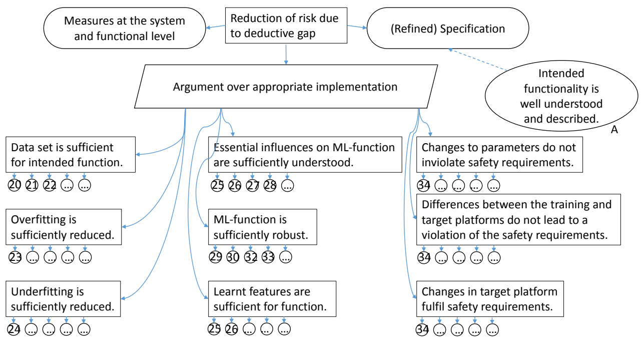
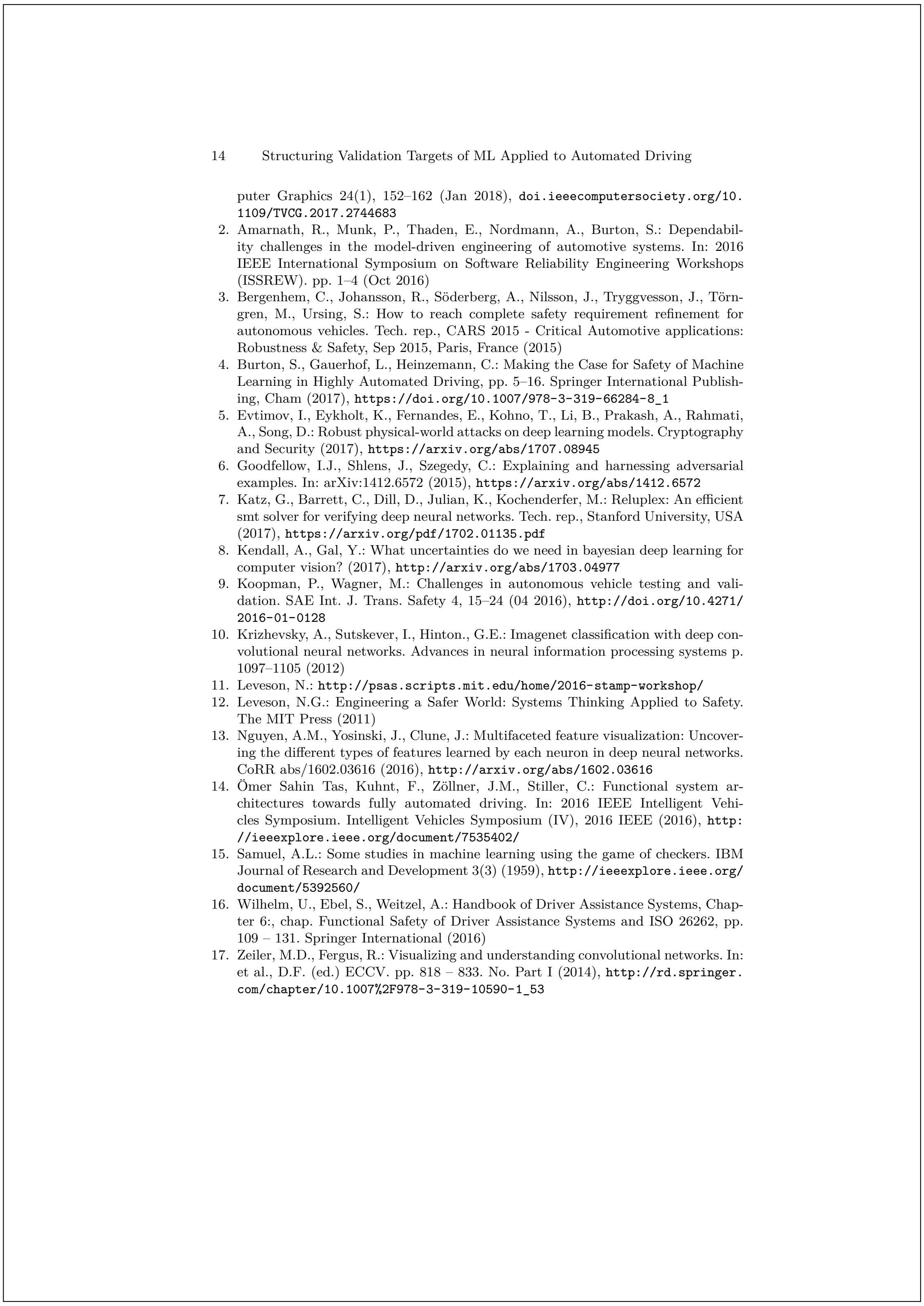

Measures at the system and functional level
Reduction of risk due to deductive gap
(Refined) Specification
Argument over appropriate implementation
Intended functionality is well understood and described.
A
Data set is sufficient for intended function.
20 21 22 ... ...
Essential influences on ML-function are sufficiently understood.
25 26 27 28 ...
Changes to parameters do not inviolate safety requirements.
34 ... ...
Overfitting is sufficiently reduced.
23 ... ...
ML-function is sufficiently robust.
29 30 32 33 ...
Differences between the training and target platforms do not lead to a violation of the safety requirements.
34 ... ...
Underfitting is sufficiently reduced.
24 ... ...
Learnt features are sufficient for function.
25 26 ... ...
Changes in target platform fulfil safety requirements.
34 ... ...

Fig. 3. GSN for the Goal Reduction of risk due to deductive gap. The following evidence is suggested to support the claims: 20) evaluation of transfer of requirements to ML-specific requirements, 21) evaluation of attribute distribution within training, test and validation data sets, 22) independence from unintended object relations, 23) overfitting measures (pretraining on diverse data set, regularisation methods, Dropout or DropConnect, data augmentation), 24) underfitting measures, 25) feature visualization, 26) correlations to features, 27) adaptation of confidence level, 28) uncertainty calculation, 29) evaluation of tolerance against distributional shift, 30) adversarial attacks, 31) integrity test of ML function, 32) statistical evaluation, 33) evaluation of faulty inputs, 34) verification specification for any changes

ness of these methods and measures must still be evaluated in detail. Therefore, further research on the contribution of each method and measure to the validation targets of automated driving is necessary. We argue that only with an industry consensus on an established set of methods, can a convincing and accepted argument for the safety of automated driving be made. This includes an agreement on a sufficient ”weight of evidence” and abort criteria for each validation activity, which has not yet been reached [2]. As part of future work we aim to systematically evaluate the effectiveness and contribution of methods and measures discussed in this paper to derive and structure validation targets and evidence for series development projects. This includes the question how the effectiveness of the individual methods and measures can be measured and demonstrated based on quantifiable key performance indicators.

References

1. Alsallakhl, B., Jourabloo, A., Ye, M., Liu, X., Ren, L.: Do convolutional neural networks learn class hierarchy? IEEE Transactions on Visualization & Com-



puter Graphics 24(1), 152–162 (Jan 2018), `doi.ieeecomputersociety.org/10.1109/TVCG.2017.2744683`

2. Amarnath, R., Munk, P., Thaden, E., Nordmann, A., Burton, S.: Dependability challenges in the model-driven engineering of automotive systems. In: 2016 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). pp. 1–4 (Oct 2016)
3. Bergenhem, C., Johansson, R., Söderberg, A., Nilsson, J., Tryggvesson, J., Törngren, M., Ursing, S.: How to reach complete safety requirement refinement for autonomous vehicles. Tech. rep., CARS 2015 - Critical Automotive applications: Robustness & Safety, Sep 2015, Paris, France (2015)
4. Burton, S., Gauerhof, L., Heinzemann, C.: Making the Case for Safety of Machine Learning in Highly Automated Driving, pp. 5–16. Springer International Publishing, Cham (2017), `https://doi.org/10.1007/978-3-319-66284-8_1`
5. Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati, A., Song, D.: Robust physical-world attacks on deep learning models. Cryptography and Security (2017), `https://arxiv.org/abs/1707.08945`
6. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: arXiv:1412.6572 (2015), `https://arxiv.org/abs/1412.6572`
7. Katz, G., Barrett, C., Dill, D., Julian, K., Kochenderfer, M.: Reluplex: An efficient smt solver for verifying deep neural networks. Tech. rep., Stanford University, USA (2017), `https://arxiv.org/pdf/1702.01135.pdf`
8. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? (2017), `http://arxiv.org/abs/1703.04977`
9. Koopman, P., Wagner, M.: Challenges in autonomous vehicle testing and validation. SAE Int. J. Trans. Safety 4, 15–24 (04 2016), `http://doi.org/10.4271/2016-01-0128`
10. Krizhevsky, A., Sutskever, I., Hinton., G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems p. 1097–1105 (2012)
11. Leveson, N.: `http://psas.scripts.mit.edu/home/2016-stamp-workshop/`
12. Leveson, N.G.: Engineering a Safer World: Systems Thinking Applied to Safety. The MIT Press (2011)
13. Nguyen, A.M., Yosinski, J., Clune, J.: Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. CoRR abs/1602.03616 (2016), `http://arxiv.org/abs/1602.03616`
14. Ömer Sahin Tas, Kuhnt, F., Zöllner, J.M., Stiller, C.: Functional system architectures towards fully automated driving. In: 2016 IEEE Intelligent Vehicles Symposium. Intelligent Vehicles Symposium (IV), 2016 IEEE (2016), `http://ieeexplore.ieee.org/document/7535402/`
15. Samuel, A.L.: Some studies in machine learning using the game of checkers. IBM Journal of Research and Development 3(3) (1959), `http://ieeexplore.ieee.org/document/5392560/`
16. Wilhelm, U., Ebel, S., Weitzel, A.: Handbook of Driver Assistance Systems, Chapter 6:, chap. Functional Safety of Driver Assistance Systems and ISO 26262, pp. 109 – 131. Springer International (2016)
17. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: et al., D.F. (ed.) ECCV. pp. 818 – 833. No. Part I (2014), `http://rd.springer.com/chapter/10.1007%2F978-3-319-10590-1_53`

Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings

Lydia Gauerhof[1,2]( ), Richard Hawkins[2]( ), Chiara Picardi[2], Colin Paterson[2], Yuki Hagiwara[1], and Ibrahim Habli[2]

[1] Corporate Research Robert Bosch GmbH, Renningen, Germany
lydia.gauerhof@de.bosch.com
[2] University of York, York, UK
richard.hawkins@york.ac.uk

Abstract. Machine Learnt Models (MLMs) are now commonly used in self-driving cars, particularly for tasks such as object detection and classification within the perception pipeline. The failure of such models to perform as intended could lead to hazardous events such as failing to stop for a pedestrian at a crossing. It is therefore crucial that the safety of the MLM can be proactively assured and should be driven by explicit and concrete safety requirements. In our previous work, we defined a process that integrates the development and assurance activities for MLMs within safety-related systems. This is used to incrementally generate the safety argument and evidence. In this paper, we apply the approach to pedestrian detection at crossings and provide an evaluation using the publicly available JAAD data set. In particular, we focus on the elicitation and analysis of ML safety requirements and how such requirements should drive the assurance activities within the data management and model learning phases. We explain the benefits of the approach and identify outstanding challenges in the context of self-driving cars.

Keywords: Machine Learning · Safety argument · Self-driving car · Safety assurance process

1 Introduction

The assurance of safety-related systems which utilise Machine Learnt Models (MLMs) can only be achieved when arguments concerning the safety of the MLM are provided in the context of the overall system into which the model is deployed. For safety-related applications, the performance of the model is just one aspect that may be of interest; we must also take a much broader view of which aspects are important to assure the safety of the MLM. These aspects should be defined in the form of explicit Machine Learning (ML) safety requirements and should drive the way in which the MLM is trained and verified, with a particular focus on the quality and suitability of the training and verification data sources.

© Springer Nature Switzerland AG 2020
A. Casimiro et al. (Eds.): SAFECOMP 2020, LNCS 12234, pp. 197–212, 2020.
https://doi.org/10.1007/978-3-030-54549-9_13

In [15] we introduced a process for generating assurance arguments for MLMs. This process integrates development and assurance activities and can be used to incrementally generate the safety assurance argument and evidence that can be used to form a safety case for the MLM within the safety related context. We also described the structure of such arguments in the form of safety argument patterns [15]. Although some simple illustrative examples were provided, the details of how to implement the process activities, and the nature of the evidence that is generated were not provided. This paper seeks to address this by considering the safety-related automated driving scenario of a self-driving car approaching a pedestrian crossing. For this scenario we use a MLM for detection of pedestrians at the crossing that is trained on a publicly available dataset [17]. In particular, by considering this credible scenario and its associated safety implications, the primary contribution of the paper is that it shows how safety requirements can be systematically and traceably generated and refined across the different lifecycle phases of the MLM, particularly focussing on the data management and model learning requirements.

The rest of this paper is structured as follows. Section 2 provides an overview of our MLM safety assurance process. In Sect. 3 we describe the autonomous driving scenario that we used for our experiment and introduce the safety requirements for the system. Section 4 details the ML requirements that we derived for the scenario. Section 5 assesses the degree to which these requirements are satisfied for the data management and model learning stages of the lifecycle respectively. Section 6 discusses related work, draws conclusions from the paper and discusses our future work.

2 Model Learning Safety Assurance Process

The process we developed for assuring the safety of MLMs was presented in [15]. We split the ML lifecycle into five stages: requirements elicitation, data management, model learning, model verification and model deployment. Traditionally ML development has focused on data collection and model performance. For safety-related systems, a much broader view of ML development is required. In particular, the requirements elicitation stage must ensure that the ML requirements reflect the intent of the broader system-level safety requirements [9]. The model verification stage must provide an independent check that the requirements are satisfied and this must be particularly focused on the verification of explicit safety requirements. The model deployment stage must ensure that the learnt model will be acceptably safe when integrated into the larger system. To ensure that each lifecycle stage provides what is required to support a safety case, we can define a set of desired properties (desiderata) for each stage. It is important to have a clear and sufficient set of desiderata. For the work reported in this paper, we have used the assurance desiderata proposed by Ashmore et al. in [1].

To ensure the desiderata are satisfied, specific ML safety requirements must be specified for each lifecycle stage. This is the focus of this paper. These ML requirements must relate to the specific safety requirements determined for the system into which the MLM will be deployed. The relationship between safety requirements at a system level and detailed ML requirements is not always obvious. For example, a safety requirement may define the need to identify all stop signs in an urban environment in sufficient time for the vehicle to stop comfortably. Turning this into specific and meaningful ML requirements relating to desiderata such as data coverage, model robustness or model accuracy is challenging and rarely discussed in a way that is justifiably traceable to system safety requirements. This paper describes how this may be done for a credible automotive scenario, focussing on ML safety requirements for data management and model learning. As part of a safety case, it must be demonstrated that the defined ML safety requirements are met. We discuss the activities that may be performed during the ML lifecycle to generate evidence to support this.

3 Pedestrian Detection at Crossings Scenario: Vehicle-Level Safety Requirements

We consider a MLM that is being used to identify pedestrians at pedestrian crossings so that an autonomous vehicle is able to stop safely. We consider that a car (the Ego vehicle) is driving autonomously in an urban environment and is approaching a crossing. We can specify a safety requirement on the Ego vehicle as follows:

Ego Shall Stop at the Crossing If a Pedestrian is Crossing
At this level the safety requirement is defined for the vehicle as a whole. It is important to note that this safety requirement would apply to the vehicle irrespective of the use of ML as part of the implementation. Based on system level safety analysis, other safety requirements could be identified (such as that the Ego vehicle should not stop unnecessarily at a crossing) but we do not consider those within this paper.

In order to elicit safety requirements for the MLM it is first necessary to identify the safety requirements that apply to the relevant system component, in this case the object detection component. The safety process decomposes the system level safety requirement to the different components of the Ego vehicle. This takes account of the proposed system architecture for the vehicle as well as the relevant operating scenarios and operating environment as discussed below.

Ego is able to sense the environment using a Bosch stereo video camera [12] that is fitted above the rear view mirror. The camera has an image size of 1280×960 pixels and a frame rate of 30 images per second. The images are sent to the object detection component that identifies pedestrians in the images and creates bounding boxes around each pedestrian. Figure 1 shows an example of an image in which all pedestrians in the scene were successfully identified. These are indicated by the green bounding boxes in the image. In this case, the image has also been annotated with white bounding boxes which show the ground truth. This indicates that even though all objects were successfully detected errors in the bounding boxes still remain. By contrast, Fig. 2 shows an image from the same crossing in which there are several identification errors in the object detection, with pedestrians who were not spotted by the object detection indicated by blue boxes in the scene.

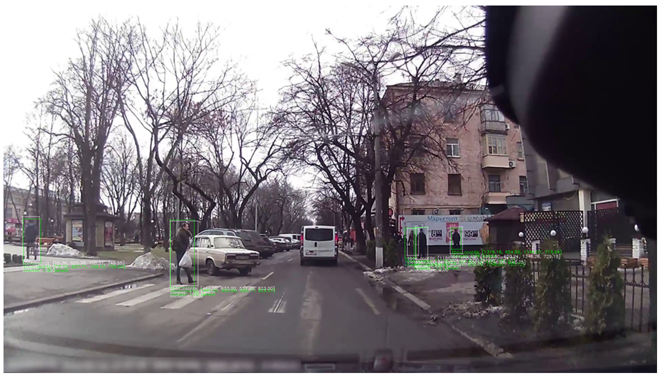


Fig. 1. An ideal pedestrian detection at crossings. (Color figure online)

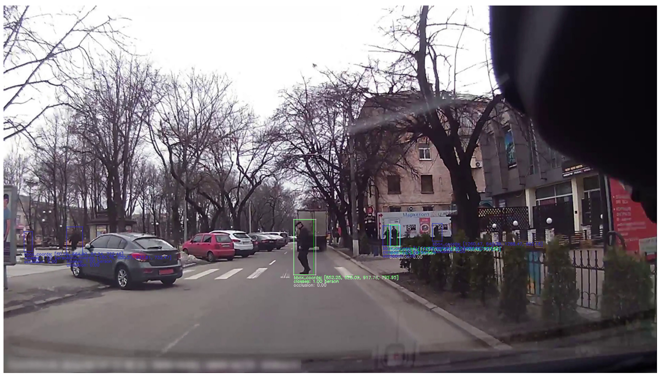


Fig. 2. An example of missed detections at crossings. (Color figure online)

It is crucial that the context within which the vehicle is expected to function is clearly and explicitly specified. For road vehicles this is normally done through the specification of the Operational Design Domain (ODD) [7]. J3016 defines an ODD as "operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics" [19].

One of the reasons for specifying the ODD is to reduce the complexity of the input space. For instance, particular geographical areas and country-specific circumstances, such as traffic signs, can be excluded. Also weather conditions such as snow, and time of day such as night, may be excluded, meaning that Ego would not operate autonomously under such conditions. Measures are put in place to ensure operation does not occur under the excluded conditions [5]. There are a number of approaches to structuring the ODD, such as equivalence classes [18]. There are also a number of ODD ontologies that have been suggested [7,11].

In the driving scenario in this paper, the ODD specifies that Ego operates on roads in the UK and in daylight, and that the weather conditions may be variable. In order to make our scenario concrete, we assume that pedestrians will only cross the road at the crossing, so we do not here consider pedestrians stepping off the pavement into the road.

Based on this we are able to specify safety requirements on the object detection component. This component is implemented in our example system using an MLM, in this case classification using a Convolutional Neural Network (CNN) based on SqueezeNet and localisation based on a Region Proposal Network (RPN). It is important to note however that at this point, the safety requirements could apply equally to the component whether it was a MLM or a more traditional software component.

To elicit the safety requirements we first consider the performance required of the object detection in order to satisfy the high-level safety requirement. Table 1 defines three performance related requirements. The justification for these requirements is provided below.

Table 1. Performance and robustness requirements for object detection.

	Performance
RQ1:	When Ego is 50 m from the crossing, the object detection component shall identify pedestrians that are on or close to the crossing in their correct position
RQ1.1:	In a sequence of images from a video feed any object to be detected should not be missed more then 1 in 5 frames
RQ1.2:	Position of pedestrians shall be determined within 50 cm of actual position
	Robustness
RQ2:	The object detection component shall perform as required in all situations Ego may encounter within the defined ODD
RQ3:	The object detection component shall perform as required in the face of defined component failures arising within the system

For RQ1, 50 m is specified as this is the minimum distance at which a decision to stop must be made if Ego is to stop comfortably at the maximum assumed speed. Stopping safely at a crossing requires consideration of this comfortable braking distance for the Ego vehicle; it would not be acceptable to brake excessively for pedestrians. The maximum comfortable braking distance will depend upon the speed of Ego and the road surface conditions. We assume for this scenario that comfortable braking loses roughly 20 kph per second on a damp road, so if Ego is travelling at 60 kph in the urban area it will take around 50 m to stop comfortably. This requires that Ego has sufficient confidence in the identification of pedestrians at 50 m, prior to this point Ego will be detecting the possible presence of pedestrians, however the uncertainty in those identifications may be relatively high.

RQ1 assumes that any pedestrian close to the crossing is intending to cross. This is certainly a conservative assumption that may result in some unrequired stopping, but is made here to simplify the scenario. In practice this could be mitigated through trajectory prediction for pedestrians (so for example pedestrians close to, but moving away from, the crossing would be rejected). It is taken that any pedestrian within one metre of the crossing is considered to be close to the crossing for the purposes of this scenario. Any pedestrians further away than this are assumed to be not intending to cross prior to Ego arriving at the crossing.

RQ1.1 and RQ1.2 further refine RQ1 by considering how good the performance of pedestrian identification and positioning needs to be in the context of the high-level system safety requirement and the system architecture. RQ1.1 is based upon the frame rate of the video feed as described above, and considers the fact that the ML model is deployed to a pipeline in which computational power is limited. As such the model may be unable to identify all objects in the scene for every frame at run-time. However the frame rate is such that the subsequent component into which the output of object detection is fed will ignore single frame changes in detections. RQ1.2 is based upon an assessment that 50 cm discrepancy in position provides a sufficient safety margin for pedestrians.

In addition to requirements on performance, it is also necessary for the performance of the object detection to be robust to the different situations that Ego may encounter. Table 1 defines two requirements relating to robustness. RQ2 is justified on the basis that if a situation that Ego encounters is outside of the defined ODD then the system will revert to a fail-safe or a manual drive mode (it is not required for object detection to cope with such situations). The safety of such transitions would be handled at the vehicle level. RQ3 acknowledges that the system components cannot be assumed to always perform perfectly. Object detection must therefore be able to cope with some defined failures or degradation. It should be noted that any failures in other system components that are not specified or are unanticipated must still be dealt with, but this would be done as part of the vehicle level safety case.

As the object detection is implemented using a MLM, these safety requirements on object detection must be interpreted to be meaningful for ML to enable assurance of the MLM to be demonstrated. In the next section we describe how ML safety requirements are derived.

4 ML Safety Requirements Elicitation

In order to create a safety argument for the MLM, it is necessary to specify concrete and meaningful ML safety requirements, i.e. traceable to the vehicle and component-level safety requirements as discussed in Sect. 3. That is, the ML requirements must be sufficient to ensure that the safety requirements identified in Sect. 3 are satisfied. The ML safety requirements are defined with a consideration of each phase of the ML lifecycle and the identified desiderata for each phase. In this paper we focus on the requirements for the data management and model learning phases.

Tables 2, 3 and 4 show the ML requirements that we have derived for these phases of the lifecycle. The tables enumerate requirements for each of the identified desiderata. For the data management phase, the desiderata we use are that the data should be relevant (Table 2), complete (Table 3), accurate (Table 4), and balanced (Table 4). These desiderata are consistent with the work of Ashmore et al. in [1] where the desiderata are discussed in more detail. The ML requirements reflect these ML desiderata within the context of the safety requirement we have identified for object detection in our scenario. A justification for these requirements is provided below.

Table 2. ML requirement elicitation for the **Relevant** desiderata of the **Data Management** lifecyle phase.

RQ4:	All data samples shall represent images of a road from the perspective of a vehicle
RQ5:	Crossings included in data samples shall be of a type found on UK roads
RQ6:	Pedestrians included in data samples shall be of a type that may use crossings on UK roads
RQ7:	The format of each data sample shall be representative of that which is captured using sensors deployed on the ego vehicle
RQ8:	Each data sample shall assume sensor positioning which is representative of that be used on the ego vehicle

If we first consider the requirements relating to the 'Relevant' desiderata, we must specify requirements that define which data is relevant to the safety requirements. Any data that is not relevant should be excluded from the data set. In order to have relevance in this context, the data sample must be an image that features a road as it may appear to Ego vehicle, and where this includes features of interest these should be relevant to the operational domain. In this case the features of interest are crossings and pedestrians. Relevant images would be expected to include some or all of these features. RQ4 to RQ6 capture this requirement for relevant data samples.

In addition the format of each image must be relevant. Since we understand the way in which images will be captured on the Ego vehicle, we can identify

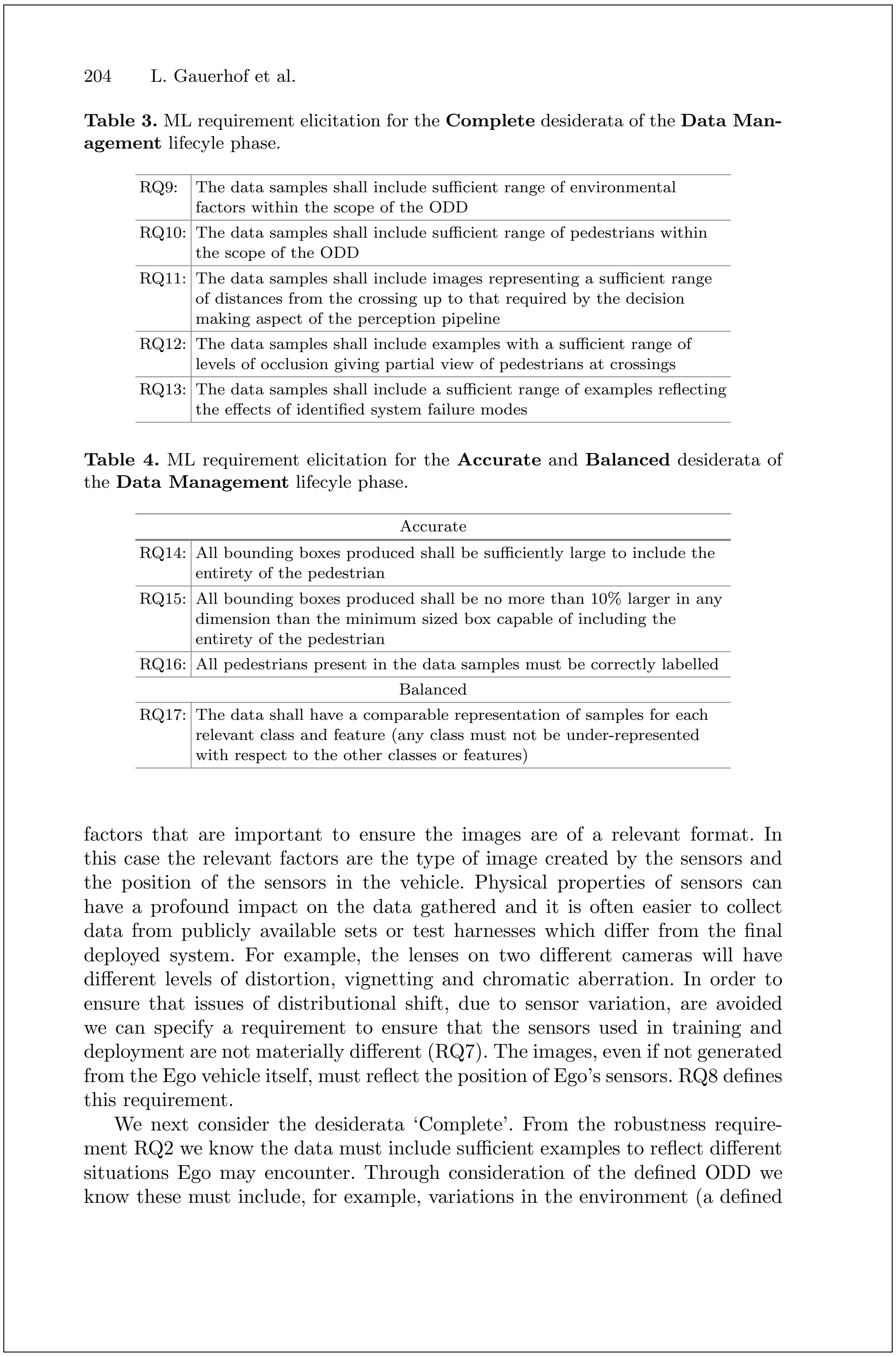


Table 3. ML requirement elicitation for the **Complete** desiderata of the **Data Management** lifecyle phase.

RQ9:	The data samples shall include sufficient range of environmental factors within the scope of the ODD
RQ10:	The data samples shall include sufficient range of pedestrians within the scope of the ODD
RQ11:	The data samples shall include images representing a sufficient range of distances from the crossing up to that required by the decision making aspect of the perception pipeline
RQ12:	The data samples shall include examples with a sufficient range of levels of occlusion giving partial view of pedestrians at crossings
RQ13:	The data samples shall include a sufficient range of examples reflecting the effects of identified system failure modes

Table 4. ML requirement elicitation for the **Accurate** and **Balanced** desiderata of the **Data Management** lifecyle phase.

	Accurate
RQ14:	All bounding boxes produced shall be sufficiently large to include the entirety of the pedestrian
RQ15:	All bounding boxes produced shall be no more than 10% larger in any dimension than the minimum sized box capable of including the entirety of the pedestrian
RQ16:	All pedestrians present in the data samples must be correctly labelled
	Balanced
RQ17:	The data shall have a comparable representation of samples for each relevant class and feature (any class must not be under-represented with respect to the other classes or features)

factors that are important to ensure the images are of a relevant format. In this case the relevant factors are the type of image created by the sensors and the position of the sensors in the vehicle. Physical properties of sensors can have a profound impact on the data gathered and it is often easier to collect data from publicly available sets or test harnesses which differ from the final deployed system. For example, the lenses on two different cameras will have different levels of distortion, vignetting and chromatic aberration. In order to ensure that issues of distributional shift, due to sensor variation, are avoided we can specify a requirement to ensure that the sensors used in training and deployment are not materially different (RQ7). The images, even if not generated from the Ego vehicle itself, must reflect the position of Ego's sensors. RQ8 defines this requirement.

We next consider the desiderata 'Complete'. From the robustness requirement RQ2 we know the data must include sufficient examples to reflect different situations Ego may encounter. Through consideration of the defined ODD we know these must include, for example, variations in the environment (a defined range of lighting and weather conditions), and in pedestrians (a defined range of ages, sizes, numbers of people and variations in gait and pose). It should be noted that an explicit enumeration of the scope of such variables is particularly critical when using MLMs in order to ensure robustness can be achieved. Experience has shown us that complex ML models can become over reliant on features in the image (over-fitting) if insufficient variation in those features is present in the data. By ensuring that a range of pedestrian features are present in the data sets we are less likely to produce models which fail to perform appropriately in the real world. RQ9 and RQ10 capture these requirements with referenced to the ODD, it is crucial therefore that the ODD is clearly documented and validated as part of the vehicle safety process. As well as exploring the scope of the ODD to consider different situations, we must also consider the impact on the images of the distance of Ego from the crossing (affecting the size of image features), and the possibility of occlusions in the image (we have discussed these effects in more detail in [2]. RQ11 and RQ12 address this issue.

From the robustness requirement RQ3 it has already been identified that the object detector must perform acceptably in the face of system failures. We acknowledge that the performance of a sensor will degrade over time, for example a camera lens will become scratched. Since this is generally unavoidable we must be confident that the performance of the object detection is not impacted by normal wear and tear. This means that the data used in the ML lifecycle must include sufficient examples that reflect the effects of these system failures on the images that are obtained. The relevant failures must be identified through failure analysis of the system (for example this could be linked to the outputs of an FMEA). RQ13 is specified to address this issue.

Another desiderata that must be considered is ‘Accurate’. The performance of MLMs is highly dependent on the quality of the data from which they learn and as such all labelling should be accurate. The performance requirement RQ1.2 specifies a performance requirement on the prediction of the pedestrian’s position. In order to assess this performance it is necessary to compare model predictions with the ground truth labels encoded in the training and testing data sets. RQ14 is therefore specified to ensure that the bounding box added to the dataset contains the whole of the pedestrian. If any part of the pedestrian, for example an arm or a leg, were not included inside the bounding box then when the model performance were assessed with reference to the bounding box a model could be deemed to meet the performance requirements when it actually breached the 50 cm required by RQ1.2. Whilst this requirement specifies a minimum size for the bounding box, it does not consider a maximum size. It would be possible to meet RQ14 by creating very large boxes around every pedestrian, however this is likely to make the system unusable as free space is essentially identified as containing a pedestrian. RQ15 addresses this issue by specifying a limit on the size of the bounding box.

The performance requirement RQ1.1 may be interpreted as an ML requirement to avoid false negatives. This leads to a requirement on the accuracy of the training data. The training data is labelled (by a human) to identify the pedestrians in each image. Manual labelling of data is error prone and drawing bounding boxes in particular is difficult. If the images are labelled incorrectly such that the pedestrians are not identified in the image then this can lead to false negatives in the output of the MLM as well. RQ16 is specified to address this.

Finally RQ17 addresses the desiderata 'Balanced'. The requirements have already specified the need for relevance and coverage in the data, it is also important that certain features are not over or under represented in the data set. Again the relevant classes and features can be identified through consideration of the ODD.

Having defined explicit ML safety requirements it is then necessary to demonstrate that the ML requirements are satisfied. In Sect. 5 we discuss whether the data we used in this experiment meet the ML safety requirements and whether additional activities are required to support a safety case. Section 5.2 then discusses this for the model that is learnt.

5 Satisfying ML Safety Requirements

In order to investigate the sufficiency of the requirements defined in this paper we considered an experimental object detection MLM consisting of a CNN trained using the JAAD dataset [17]. In this section the data management and model learning for this MLM is assessed against the defined requirements to determine whether the requirements are satisfied. This highlighted areas where the MLM is insufficient from a safety assurance perspective, and identified additional assurance activities that would be required. This highlights the key role of an explicit elicitation of ML safety requirements in assuring MLMs.

5.1 Assessing the Data Management Safety Requirements

In this section, we discuss each data requirement presented in Tables 2, 3 and 4 with reference to the JAAD dataset used in our experiment.

RQ4–5: In the dataset there are 25 videos relative to pedestrian crossing at designated and signalised crossings. For each of these videos approximately 82,000 image samples can be extracted. The recordings were done during 240 h of driving across several locations in North America and Eastern Europe. Even if some of the crossings could be considered similar between Eastern Europe and UK (e.g. zebra and pegasus crossings), the data does not meet this requirements because UK locations are not included in the recording and therefore not all UK pedestrians crossings types are considered. In particular it can be easily noted that Pelican crossings are not included in the data. Augmenting the data by synthesising missing images can partially solve the problem, but the data samples generated must be very close to real world images. A better solution could be to undertake additional data collection in different UK crossing locations.

RQ6: When considering the JAAD dataset, we see that many classes of pedestrian are included, e.g. examples of children are included as is a man pushing a buggy. There are however some relevant omissions from this including people with disabilities, and people with different colour skin or ethnicity. When considering if there are any particular characteristics of UK pedestrians it seems important given the UK climate to ensure there are images of people carrying umbrellas or wearing waterproof jackets. These are not found in the data. Therefore the dataset could not be said to meet this requirement as more data would need to be collected that included the missing categories.

RQ7: The cameras used for the recording of the dataset are describe in [16]. The resolution of the three cameras are compatible with the one deployed in the ego vehicle (see Sect. 3). Consequently, this requirement can be considered satisfied.

RQ8: The camera recording the data used is positioned inside the car below the rear view mirror as described in [16]. In the ego vehicle the camera is mounted inside the car above the rear view mirror. Although the position is not exactly the same of the ones used for the data recording the distance is not significant and as such the requirement is satisfied.

RQ9–10: The data represents some different weather conditions (e.g. snow and rain). They do not consider different positions of the sun or different daytime lighting (e.g. sunset). Limited visibility weather conditions like fog are also not included, even though this is part of the ODD. The data includes pedestrians of different ages and height, as well as different walking speeds. No running pedestrians are included however. Although there are a sufficient range of examples for some features, for others the data is found to be lacking. Augmentation techniques can be applied to address this, for example by varying the colour of pixel or the orientation of pedestrians to the camera as done in previous work (e.g. [6]). In particular, Zhang and colleagues [20] described a method, through the use of a Generative Adversial Network (GAN), to synthesize scenes for autonomous driving simulating different weather conditions and then different lighting conditions. Further, possible evidence for supporting the argument in order to satisfy the requirements can be represented by performance graphs showing the difference between original and augmented data and how the different features included influence the performance. The data include some busy crossings that have groups of up to a maximum of 11 people. The performance of the MLM in identifying pedestrians when in groups compared to individuals could be used as evidence of this requirement. If performance is seen to be worse for groups, then more data samples for groups of people should be included.

RQ11: In most of the images included in the dataset, the pedestrians are very close to the car so do not respect the distance necessary for the pipeline decision making (in excess of 50 m). The dataset would therefore be inappropriate against this requirement.

RQ12: There is partial occlusion of pedestrians in some of the data samples. For example, some pedestrians are occluded by gates or by the car in front of the Ego vehicle. Again, the number of occluded data samples could be increased through synthesis. For example artificial masking of pedestrians could be used to help meet the requirement.

RQ13: Data samples derived from identified failures in the system are not present in the dataset. Also the classifier is not tested with adversarial attacks. For these reasons the requirements are not satisfied. In order to satisfy these requirements failures need to be identified and recorded in a report that can be used as evidence to support the argument. After failures are identified, corresponding data samples need to be added to the data set.

Req.14–16: While the process used to generate the dataset is described in [16], there is limited information regarding the generation of bounding boxes. Piotr's annotation toolbox [8] is used to define the bounding boxes and annotate the images. However, there is no information regarding the process to ensure that these are correct with respect to ground truth. The accuracy of the labelling is a function of the skills of the individuals undertaking the task and the validation processes used during labelling.

RQ17: Using a public dataset results in a lack of information and control in the number of data samples recorded for each feature of interest. Features that are under-represented have to be identified and possibly over-sampled in order to improve the performance of the classifier in presence of these features. Augmentation approaches can be used here, as well as other techniques for detecting and mitigating rare classes, such as [14].

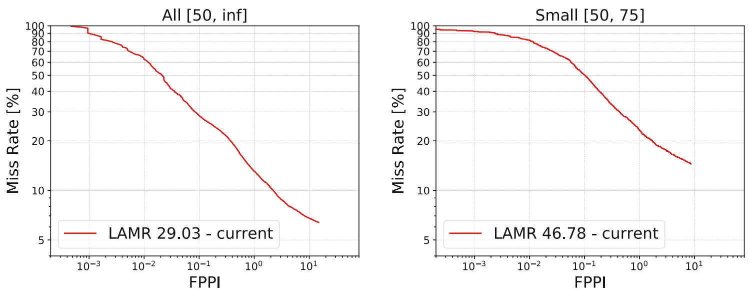
In short, a public dataset such as JAAD is not sufficient to satisfy the ML safety requirements for our scenario. This result is not unexpected, but it highlights the role of explicit ML safety requirements in both highlighting deficiencies, and identifying necessary actions. Public datasets can however be useful for an exploratory analysis in order to refine the requirements as suggested by Gelman and colleagues [10].

5.2 Assessing the Model Learning Safety Requirements

In this section we discuss the ML safety requirements that relate to the learned model itself as presented in Table 1, with reference to the model used in our experiment.

RQ1: In order to evaluate classifiers in the automotive domain it is common practice to use the log average miss rate (LAMR) [13]. Having constructed a convolution neural network as an MLM, we calculated the LAMR for images in the dataset. When considering all pedestrians larger than 50 pixels in the image, we obtain an LAMR of 29.03%. We note that for those pedestrians between 50 and 75 pixels this increases to 46.78%. These results are shown in Fig. 3a and Fig. 3b with more detail provided in Table 5.

RQ1.1: The JAAD dataset does provide a labelling which allows for each object to be tracked through frames. However, at present we do not have access to a



a) MR vs. FPPI for pedestrians larger than 50px in height.

b) MR vs. FPPI for pedestrians detection of the size 50px to 75px.

Fig. 3. Miss rate (MR) vs. false positive per image (FPPI) for pedestrian detection with different heights.

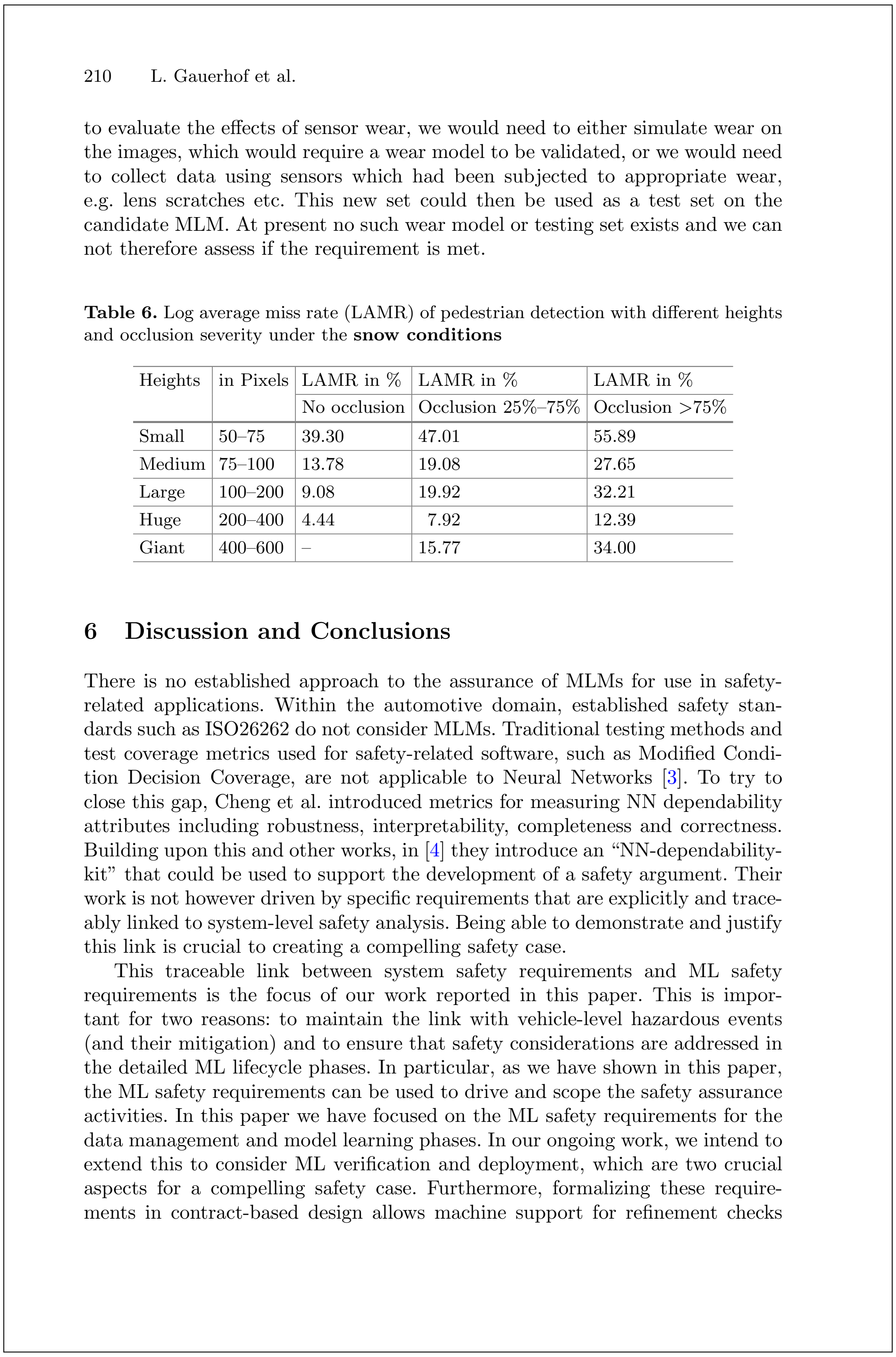
pipeline which allows us to generate evidence to evaluate whether the MLM meets this requirement. This remains as future work.

RQ2: The images in the data set only cover 5 locations with the vast majority of videos captured in one location. Some of the images included weather features, for example LAMR results are shown in Table 6 for LAMR under snow conditions. Even these cases are restricted since snow is lying on the ground, so variations such as falling snow are missing. The generation of the JAAD database used for training required considerable effort, especially in the labelling of objects within the scene. In order to assess the ability of the MLM to operate at locations other than those in the JAAD dataset would require additional data collection and significant labelling effort. Without this, it is impossible to assess if the requirement could be met using this MLM.

Table 5. Log average miss rate (LAMR) of pedestrian detection with different heights of bounding boxes and occlusion severity (the smaller, the better).

Heights	in Pixels	LAMR in %	LAMR in %	LAMR in %
		No occlusion	Occlusion 25%–75%	Occlusion >75%
Small	50–75	46.78	54.12	62.18
Medium	75–100	20.22	28.91	36.49
Large	100–200	7.96	16.14	25.72
Huge	200–400	7.47	13.18	19.05
Giant	400–600	10.76	21.18	31.03

RQ3: The JAAD videos were not captured using sensors traditionally used for autonomous vehicles. Instead, consumer video cameras were employed. In order



to evaluate the effects of sensor wear, we would need to either simulate wear on the images, which would require a wear model to be validated, or we would need to collect data using sensors which had been subjected to appropriate wear, e.g. lens scratches etc. This new set could then be used as a test set on the candidate MLM. At present no such wear model or testing set exists and we can not therefore assess if the requirement is met.

Table 6. Log average miss rate (LAMR) of pedestrian detection with different heights and occlusion severity under the **snow conditions**

Heights	in Pixels	LAMR in %	LAMR in %	LAMR in %
		No occlusion	Occlusion 25%–75%	Occlusion >75%
Small	50–75	39.30	47.01	55.89
Medium	75–100	13.78	19.08	27.65
Large	100–200	9.08	19.92	32.21
Huge	200–400	4.44	7.92	12.39
Giant	400–600	–	15.77	34.00

6 Discussion and Conclusions

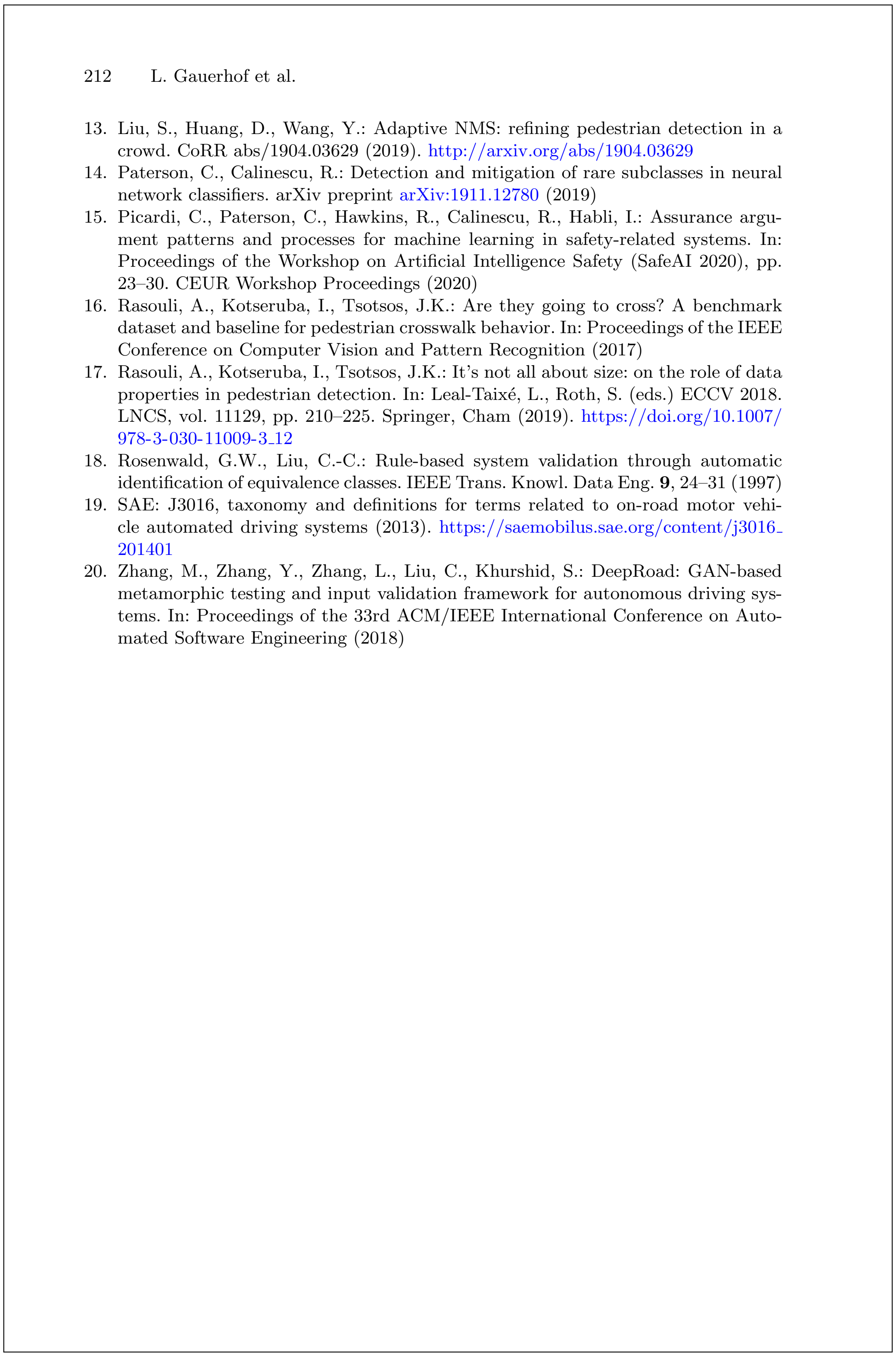
There is no established approach to the assurance of MLMs for use in safety-related applications. Within the automotive domain, established safety standards such as ISO26262 do not consider MLMs. Traditional testing methods and test coverage metrics used for safety-related software, such as Modified Condition Decision Coverage, are not applicable to Neural Networks [3]. To try to close this gap, Cheng et al. introduced metrics for measuring NN dependability attributes including robustness, interpretability, completeness and correctness. Building upon this and other works, in [4] they introduce an "NN-dependability-kit" that could be used to support the development of a safety argument. Their work is not however driven by specific requirements that are explicitly and traceably linked to system-level safety analysis. Being able to demonstrate and justify this link is crucial to creating a compelling safety case.

This traceable link between system safety requirements and ML safety requirements is the focus of our work reported in this paper. This is important for two reasons: to maintain the link with vehicle-level hazardous events (and their mitigation) and to ensure that safety considerations are addressed in the detailed ML lifecycle phases. In particular, as we have shown in this paper, the ML safety requirements can be used to drive and scope the safety assurance activities. In this paper we have focused on the ML safety requirements for the data management and model learning phases. In our ongoing work, we intend to extend this to consider ML verification and deployment, which are two crucial aspects for a compelling safety case. Furthermore, formalizing these requirements in contract-based design allows machine support for refinement checks within a component-based system [2]. We hope that this work is of benefit to both researchers and engineers and helps inform the current debate concerning the safety assurance and regulation of autonomous driving.

Acknowledgements. This work is funded by the Assuring Autonomy International Programme https://www.york.ac.uk/assuring-autonomy.

References

1. Ashmore, R., Calinescu, R., Paterson, C.: Assuring the machine learning lifecycle: desiderata, methods, and challenges (2019). http://arxiv.org/abs/1905.04223
2. Burton, S., Gauerhof, L., Sethy, B.B., Habli, I., Hawkins, R.: Confidence arguments for evidence of performance in machine learning for highly automated driving functions. In: Romanovsky, A., Troubitsyna, E., Gashi, I., Schoitsch, E., Bitsch, F. (eds.) SAFECOMP 2019. LNCS, vol. 11699, pp. 365–377. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26250-1_30
3. Cheng, C., Nührenberg, G., Huang, C., Ruess, H., Yasuoka, H.: Towards dependability metrics for neural networks. In: 2018 16th ACM/IEEE International Conference on Formal Methods and Models for System Design (2018)
4. Cheng, C.H., Huang, C.H., Nührenberg, G.: nn-dependability-kit: engineering neural networks for safety-critical autonomous driving systems. arXiv:1811.06746 (2018)
5. Colwell, I., Phan, B., Saleem, S., Salay, R., Czarnecki, K.: An automated vehicle safety concept based on runtime restriction of the operational design domain. In: 2018 IEEE Intelligent Vehicles Symposium (IV) (2018)
6. Crispell, D., Biris, O., Crosswhite, N., Byrne, J., Mundy, J.L.: Dataset augmentation for pose and lighting invariant face recognition. arXiv:1704.04326 (2017)
7. Czarnecki, K.: Operational world model ontology for automated driving systems–part 1: road structure. Waterloo Intelligent Systems Engineering Lab (WISE) Report (2018)
8. Dollár, P.: Piotr's computer vision matlab toolbox (PMT). https://github.com/pdollar/toolbox
9. Gauerhof, L., Munk, P., Burton, S.: Structuring validation targets of a machine learning function applied to automated driving. In: Gallina, B., Skavhaug, A., Bitsch, F. (eds.) SAFECOMP 2018. LNCS, vol. 11093, pp. 45–58. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99130-6_4
10. Gelman, A., Loken, E.: The garden of forking paths: Why multiple comparisons can be a problem, even when there is no "fishing expedition" or "p-hacking" and the research hypothesis was posited ahead of time. Columbia University, Department of Statistics (2013)
11. Geyer, S., et al.: Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance. IET Intell. Transp. Syst. **8**, 183–189 (2013)
12. GmbH, R.B.: Stereo video camera product data sheet. https://www.bosch-mobility-solutions.com/media/global/products-and-services/passenger-cars-and-light-commercial-vehicles/driver-assistance-systems/lane-departure-warning/stereo-video-camera/product-data-sheet-stereo-video-camera.pdf. Accessed 28 Feb 2020



13. Liu, S., Huang, D., Wang, Y.: Adaptive NMS: refining pedestrian detection in a crowd. CoRR abs/1904.03629 (2019). http://arxiv.org/abs/1904.03629
14. Paterson, C., Calinescu, R.: Detection and mitigation of rare subclasses in neural network classifiers. arXiv preprint arXiv:1911.12780 (2019)
15. Picardi, C., Paterson, C., Hawkins, R., Calinescu, R., Habli, I.: Assurance argument patterns and processes for machine learning in safety-related systems. In: Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI 2020), pp. 23–30. CEUR Workshop Proceedings (2020)
16. Rasouli, A., Kotseruba, I., Tsotsos, J.K.: Are they going to cross? A benchmark dataset and baseline for pedestrian crosswalk behavior. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
17. Rasouli, A., Kotseruba, I., Tsotsos, J.K.: It's not all about size: on the role of data properties in pedestrian detection. In: Leal-Taixé, L., Roth, S. (eds.) ECCV 2018. LNCS, vol. 11129, pp. 210–225. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11009-3_12
18. Rosenwald, G.W., Liu, C.-C.: Rule-based system validation through automatic identification of equivalence classes. IEEE Trans. Knowl. Data Eng. **9**, 24–31 (1997)
19. SAE: J3016, taxonomy and definitions for terms related to on-road motor vehicle automated driving systems (2013). https://saemobilus.sae.org/content/j3016_201401
20. Zhang, M., Zhang, Y., Zhang, L., Liu, C., Khurshid, S.: DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (2018)

Considering Reliability of Deep Learning Function to Boost Data Suitability and Anomaly Detection

Lydia Gauerhof
Corporate Research
Robert Bosch GmbH
Renningen, Germany
0000-0002-3504-0040

Yuki Hagiwara
Corporate Research
Robert Bosch GmbH
Renningen, Germany
0000-0002-5418-738X

Christoph Schorn
Corporate Research
Robert Bosch GmbH
Renningen, Germany
0000-0002-1589-7936

Mario Trapp
Fraunhofer Institute for Cognitive Systems IKS
Munich, Germany
0000-0002-4716-7372

***Abstract*—The increased demand of Deep Neural Networks (DNNs) in safety-critical systems, such as autonomous vehicles, leads to increasing importance of training data suitability. Firstly, we focus on how to extract the relevant data content for ensuring DNN reliability. Then, we identify error categories and propose mitigation measures with emphasis on data suitability. Despite all efforts to boost data suitability, not all possible variations of a real application can be identified. Hence, we analyse the case of unknown out-of-distribution data. In this case, we suggest to complement data suitability with online anomaly detection using FACER that supervises the behaviour of the DNN.**

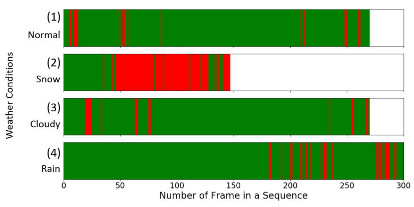
***Index Terms*—data analysis, data suitability, anomaly detection**

I. Introduction

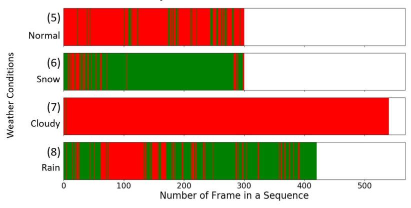
In the context of automated driving, Deep Neural Networks (DNNs) are widely used. For example, video based object detection can be implemented using a region proposal network (RPN) [1] to enable pedestrian detection (PDET) of an automated vehicle. Thereby, an increased variety and broad distribution of the training dataset improves DNN reliability. However, data amount and variety alone are not sufficient to guarantee reliable behaviour. To optimise data collection in an effective way, missing or over-sampled data characteristics shall be known as well. On one hand, data content might be relevant to the task to be learned by a DNN. If this content is under-sampled, the chances that the DNN will not perform under this condition as intended is higher, and thus, producing unsatisfactory results. On the other hand, data might include content that the DNN recognises as a pattern and learn to correlate to. If these patterns are not relevant to the task, they might cause unreliable behaviour which is also unintended. We use the term data suitability when no under-sampling of relevant content and freedom from unintended correlations are given. In this case, data reflects the intended functionality.

To analyse data regarding required properties, often data themselves are used. For example, approaches for anomaly detection utilise data to evaluate abnormal data. Moreover, data density and distribution might be investigated.

Besides analysing data properties directly, the reliability of the DNN that is trained on the corresponding data is a valuable indicator for data suitability. Consequently, error analysis of the DNN on test data can be used for collecting more suitable training data.



(a) Frame analysis of videos taken in "Plaza"



(b) Frame analysis of videos taken in "Street"

Fig. 1. Error analysis on sequences: Red and green segments refer to wrong and correct PDET predictions respectively.

In a consecutive step, we propose to complement data suitability with online anomaly detection for two reasons. First, test data only represent a subset of all possible variations of a real application. Second, input data at run-time is exposed to distributional shift and the appearance of new classes.

In our case study, we focus on a PDET function for automated driving. Our contributions are:

- We analyse DNN performance on data sequences (Fig. 1) and identify error categories.
- We perform sensitivity analysis of PDET on out-of-distribution data to exhibit remaining vulnerability.
- We emphasise data suitability in mitigation measures.
- We propose the anomaly detector FACER dependent on PDET behaviour to cover errors caused by out-of-distribution data. The novel contribution on FACER is that we apply it on object detection instead of a classifier.

II. Related Work

Many DNNs are trained on a given dataset which is a good starting point for training. Different data augmentation techniques have been introduced, to improve training results. Data augmentation can be regarded as measure to improve data suitability, as it might help to mitigate over-fitting on unintended correlations and under-sampling of relevant content.

Geometric transformations are frequently used to increase training set size in image classification [2]. For example, the horizontally flipped version of an image can be considered realistic in many but not all cases. Moreover, different types of noises can be added on data to increase various notions of robustness [3]. Hendrycks *et al.* [4] benchmark neural networks robustness to common corruptions and perturbations. Different noise types are considered to reflect physical effects that appear on real data. However, most augmentation approaches approximate effects without considering all details, *e.g.*, illumination, that are present in reality.

Another option to augment the training set is the use of synthetic data. Generative adversarial models [5] exploit the fact that high-dimensional image patterns can be encoded in a low-dimensional latent space. Recent developments of generative adversarial models focus on the generation of realistic and diverse images [6], such as the Reverse Variational Autoencoder [7]. Despite these advances, the image embedding in a latent space and image generation remain hardly comprehensive to human understanding. Consequently, these approaches should be combined with explainable and trustworthy methods to increase confidence in the methodology and data suitability.

There are also other methods to generate synthetic data. Different 3D rendering approaches [8], [9] enable photo-realistic data. Despite this success, it has to be evaluated if training on synthetic data leads to comparable DNN behaviour as with real data. Some artefacts might be different for synthetic and real data. Variability caused by imperfection of reality is not present in synthetic data.

When it comes to safety-critical functions, both reliability of the DNN and data suitability have to be verified. Plain data augmentation without reliability analysis of the DNN may fail to cover all relevant content for training the intended functionality. Besides that, data augmentation does not completely ensure the freedom from unintended correlations. In contrast to previous approaches, we propose to feed back insights from DNN reliability analysis to boost data suitability. Moreover, we argue that handling out-of-distribution data should be a subsequent step.

III. Case Study

We perform a case study in which we evaluate reliability of a PDET function based on safety requirements (RQ) proposed by Gauerhof *et al.* [10]. Our PDET is implemented using the SqueezeNet DNN architecture [11], combined with an RPN [1], trained on the dataset Joint Attention for Autonomous Driving (JAAD) [12]–[14] during development, and we assume no online training in the field.

To leverage our knowledge about *training data suitability*, we evaluate reoccurring errors in Section III-A based on RQ1.1 [10]: "In a sequence of images from a video feed, any object to be detected should not be missed more than 1 in 5 frames." By manually extracting reoccurring error categories, we find data content that is not well learnt by DNN. This data content might be not intuitive to humans. By doing so, we want to know which data content is relevant and shall not be under-represented as specified by the safety requirement for the data management RQ17 [10]: "The data shall have a comparable representation of samples for each relevant class and feature (any class must not be under-represented with respect to the other classes or features)."

Despite all these efforts, there will always be data within the defined operational design domain (ODD) during run-time that we have not anticipated. Therefore, we evaluate *sensitivity to out-of-distribution data* of the PDET in Section III-B.

A. Identification of Error Categories in Data Sequences

Time dependent, sequential data, where a physical continuation is given, can change content-wise only to a limited extent. Our aim is to investigate if there are reoccurring errors of the PDET function in a sequence. This allows us to identify learnt correlations, some of which cannot be intuitively identified by considering data alone. Our method is divided into two steps.

First, we extract false negative (FN) and false positive (FP) pedestrian detections from video sequences. A frame is deemed correct when all pedestrians are detected, while a wrong prediction means that there is at least an FN or an FP. We group eight JAAD videos from the test dataset with different weather conditions into two locations:

- Plaza: open space car park, mall (Fig. 1a)
- Street: buildings, sculptures, shops, traffic (Fig. 1b)

Fig. 1 reflects the occurrence of correct (green segments) and wrong (red segments) predictions for every frame in each of the eight video sequences. This first step can be easily automated to evaluate large and complicated datasets.

In the second step, erroneous sequences are manually inspected to identify weaknesses in training data suitability. The main idea of the breakdown of errors into different error categories is to provide a more in-depth understanding in regard to the FPs and FNs for improving the training dataset. For the JAAD dataset and the PDET function of our case study we identify ten error categories, shown in Fig. 2:

1) FN on occluded pedestrian
2) FN on fully visible pedestrian
3) FN on camouflaged pedestrian
4) Predicted bounding box (BB) shifted from ground truth (GT)
5) Missing labels
6) FP on random objects
7) FP on signpost, lamppost, traffic light, traffic barrier pole
8) FP on tree, tree branches
9) FP on statue
10) FP on pedestrian (double detection)

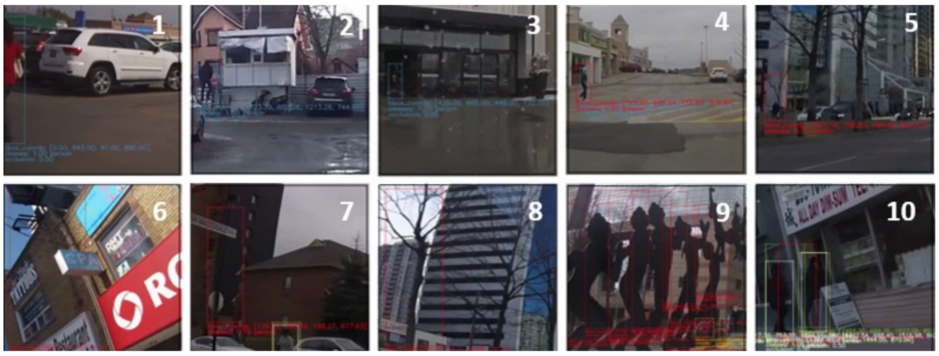


Fig. 2. Error categories: 1) FN on occluded pedestrian; 2) FN on fully visible pedestrian; 3) FN on camouflaged pedestrian; 4) Predicted BB shifted from GT; 5) Missing labels; 6) FP on random objects; 7) FP on signpost, lamppost, traffic light, traffic barrier pole; 8) FP on tree, tree branches; 9) FP on statue; 10) FP on pedestrian (double detection)

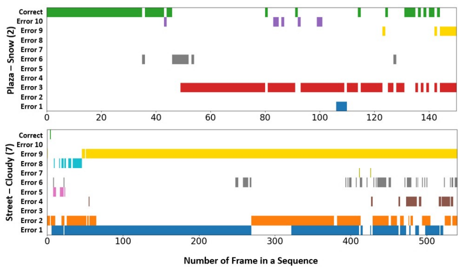
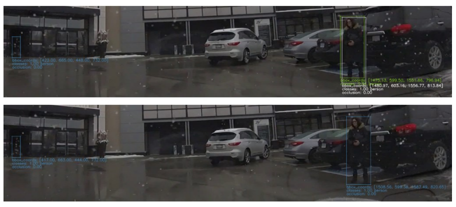
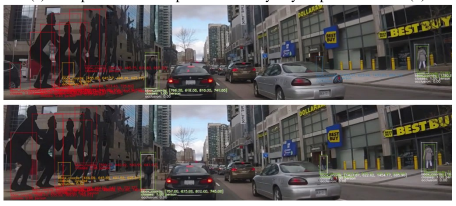


Fig. 3. Predictions evaluation on video sequences (2) and (7).



(a) Two prediction examples on a snowy day in plaza in video (2)

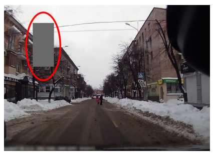


(b) Two prediction examples on a cloudy day on a street in video (7)

Fig. 4. Visualisation of predictions: Bounding boxes represent false positives (red), false negatives (blue), true positives (green), and ground truth (white).

For example in Fig. 1a, most errors, exhibited in red, are found on a snowy day (video (2)). We identified that the snowy weather condition is not the main reason for this. Most errors can be allocated to error category 3, FNs on two camouflaged pedestrians, see upper diagram of Fig. 3. We define a camouflaged pedestrian as concealed in the background such that it is visually more difficult to spot the pedestrian. This implies that the contrast between object and background is very low. First, there is a repeated FN on a camouflaged pedestrian at the entrance of the mall, see example of error category 3 in Fig. 2. Second, there is an FN on a camouflaged pedestrian detection near the vehicle. Both examples are depicted in Fig. 4a. Failing to detect at least the pedestrian near the vehicle could lead to an accident or even loss of human life.

In Fig. 1b, the street video (7) on a cloudy day has the longest streak of erroneous frames. Fig. 4b provides two samples of systematic wrong predictions from PDET. First, we observe that FPs on statues largely contribute to the errors, see error category 9 in the lower diagram of Fig. 3. Second, there is also a systematic FN on a partially occluded pedestrian on the right side of the image in the first row of Fig. 4b. This is expressed by error category 1 in the lower diagram of Fig. 3. It can also be seen that error category 2 contributed to a large extent in video (7) (see Fig. 3). This is mainly due to many far-away pedestrians on the street. In this case FN detections of occluded or smaller-sized pedestrians at the pavement of the street might be not a serious problem. This is because, vehicles would not drive towards the pavement, where most pedestrians are, under normal circumstances. Instead, the focus is on the


(a) Grey box

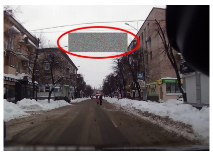

(b) Noisy box

Fig. 5. Examples of images with random boxes.

pedestrian crossing the road who is correctly recognized by the PDET function (see Fig. 4b). Furthermore, the criticality of the misidentified row of statues at the pavement might be dependent on subsequent components, such as motion predictors.

B. *Out-of-Distribution Sensitivity Analysis*

In order to gain knowledge of the PDET reliability on out-of-distribution data, we conduct a sensitivity analysis. Random grey and noisy boxes are injected on the test data to occlude certain regions of the data which do not necessarily cover pedestrians. This approach is also known as random erasing and has been originally developed as part of data augmentation to train DNNs to recognise objects with occlusion [15]. Nonetheless, we employ this technique for evaluating the sensitivity of PDET to out-of-distribution data. We use two box types, with and without noise, since they activate the feature detectors of the DNN in different ways. Fig. 5 shows an example of both grey and noisy boxes. We regard these boxes as relevant test cases, especially noisy boxes, since they could represent a noisy poster or a T-shirt with printed noise. Similar to that might be, for example, stickers on road signs which are already known as real world attacks to artificial perception systems. Table I records the log-average miss rate (LAMR) evaluated on the random erasing technique. LAMR decreases for occlusion with grey box and degrades drastically for occlusion with box filled with white noise. The local noise seems to propagate through the DNN and affect PDET results not only locally, but also globally at different positions of the image. It seems that, especially with white noise, much of the pixel information is distorted by the random erasing technique.

Hence, this provides evidence to substantiate that PDET is not robust towards noisy data. One mitigation measure could be to extend the training dataset in such a way that the required robustness is achieved. Nevertheless, there will always be data content that we have not considered or that might appear in future. To cover data that is out-of-distribution, a mitigation measure would be to implement a run-time monitor. This run-time monitor should be an anomaly detector that takes the reliability of the PDET into account.

IV. Mitigation Measures

With the insights gained from our case study, we suggest mitigation measures to increase reliability and out-of-distribution detection in the following. First, we focus on strategies for improving reliability at design time by enhancing training data suitability in Section IV-A. Second, to mitigate unreliable behaviour of DNN in the field due to out-of-distribution data, we propose an anomaly detector that is dependent on the DNN behaviour in Section IV-B.

TABLE I
Log-average miss rate (%) of PDET with random erasing approach w.r.t. pedestrians larger than 50 pixels in height

	No Box	**Grey Box**	**Noisy Box**
LAMR (%)	29.03	35.49	97.95

A. *Enhancing Training Data Suitability*

Corresponding to the identified error categories 1 to 3 and 6 to 10, data should be collected in order to enrich the training data. Consequently, chances that the DNN learns correlations to data that are relevant to the task would increase, which would in turn increase reliability. Therefore, reliability of the DNN is an essential measure for training data suitability. For example, FP on trees can be reduced by including more data with trees in training dataset, so that the DNN learns to distinguish trees from pedestrians. Lampposts and signposts shall also be enriched in the training data. This can be done in an iterative process where error sequence analysis (see Section III-A) and training data adjustment are performed in sequence until PDET reliability reaches an acceptable level.

In error category 4, the predicted BB is shifted from GT, where in most cases the predicted BB is better fitted to the actual pedestrian position than GT. Furthermore, in error category 5, labels are missing. Both error categories lead to the conclusion that labelling quality has to be increased. We expect that this also helps to diminish errors from category 10, where pedestrians are detected twice.

Moreover, meta-information in data regarding the detection criticality should put emphasis on safety-critical situations. Rare and dangerous events that might cause systematic errors should be over-sampled for training dataset to achieve reliable behaviour in these situations.

We conclude that balanced data is achieved and thus, RQ17 is satisfied when a reliable DNN behaviour under all relevant circumstances, as specified by the ODD, is achieved.

B. *Run-Time Monitor: FACER*

As a run-time monitor, to detect out-of-distribution data, we employ an anomaly detector known as **f**eature **a**ctivation **c**onsistency check**er** (FACER) [16]. Its performance is on par with other state-of-the-art methods. Until now, FACER has been applied to classifiers. In this study, we extend FACER to object detection. In contrast to many anomaly detectors that operate only on the input data, FACER takes the PDET behaviour into account by retrieving a condensed form of the intermediate neuron outputs of the DNN, called feature activation vector $\mathbf{f}_{\mathrm{act}}$, as depicted in Fig. 6. Summation over the values of each feature map in all layers of the DNN leads to the vector $\mathbf{f}_{\mathrm{act}}$ composed of feature activation values. We normalise each $\mathbf{f}_{\mathrm{act}}$ by subtracting mean and dividing by

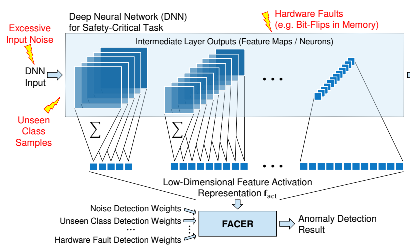

Excessive Input Noise
DNN Input
Unseen Class Samples
Deep Neural Network (DNN) for Safety-Critical Task
Hardware Faults (e.g. Bit-Flips in Memory)
Intermediate Layer Outputs (Feature Maps / Neurons)
Task Result
Low-Dimensional Feature Activation Representation $\mathbf{f}_{\text{act}}$
Noise Detection Weights
Unseen Class Detection Weights
Hardware Fault Detection Weights
FACER
Anomaly Detection Result

Fig. 6. **F**eature **a**ctivation **c**onsistency check**er** (FACER) can be trained to detect various types of anomalies in a DNN [16].

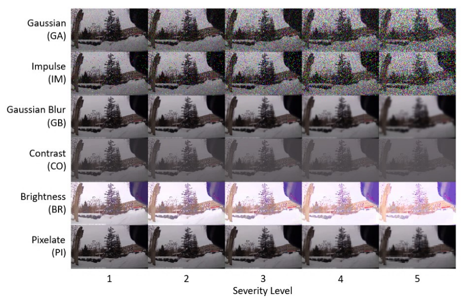


Fig. 7. Augmented data example of each noise type and increasing corruption severity from 1 to 5.

standard deviation values per element obtained over the non-anomaly $\mathbf{f}_{\text{act}}$ training set. To avoid division by zero, we add a small offset of 10^{-8} to the standard deviation. FACER predicts the output, either anomaly or non-anomaly, using a small feedforward neural network internally and $\mathbf{f}_{\text{act}}$ as input. Hence, FACER detects anomalies in the PDET feature representation. PDET retraining is not necessary to that end, but only FACER itself is trained on only binary anomaly or non-anomaly labels.

We train with the "Adam" optimizer, setting the learning rate to 0.001, $\beta^1 = 0.9$, $\beta^2 = 0.999$, and $\varepsilon = 1 \times 10^{-8}$, as suggested in [17]. FACER is trained for eight epochs with a batch size of 4000. There are 27 layers of output neurons extracted and the feature activations are concatenated in $\mathbf{f}_{\text{act}}$ with a dimension of 2966 to form the input for FACER.

Instead of starting with the previous random erasing experiment, we want to know which types of noise are more challenging to detect. This serves as a proof-of-concept that FACER is able to handle different noise types which can affect PDET reliability. Therefore, we extend our experiments to evaluate six noise types with five levels of severity (see Fig. 7).

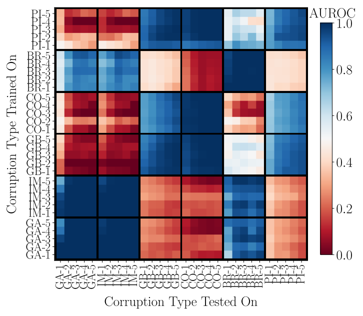


Fig. 8. AUROC values of FACER for all noise types and severity levels. Note that noise types are represented by their respective acronym and the numbers indicate the severity.

They are gaussian (GA), impulse (IM), gaussian blur (GB), contrast (CO), brightness (BR) and pixelate (PI). Similarly to Schorn and Gauerhof [16], we consider four different test cases and evaluate FACER with area under the receiver operating curve (AUROC).

a) Training and testing on a single noise type: A heat map is drawn in Fig. 8 to represent the results yielded from all noise types and all levels of severity. From the results obtained in this test case, it can be observed that certain noise types which have different properties do not perform well together.

b) Testing one noise type after training on all the others: Fig. 9 provides a heat map for an overview of the resulting AUROCs acquired. We can see that generally, FACER detects each noise as anomaly relatively well even though it has not been trained on the noise type. We also observe that FACER has difficulty differentiating brightness noise as anomalies when it is not trained. This could be due to an inherent problem that exists in data. Fig. 10 shows two examples of JAAD training data that have different degrees of brightness without being manipulated. For example, a level five brightness on the darker image could still be darker than a level one brightness on the brighter image. Therefore, it is more challenging to identify brightness anomalies.

c) Training on all noise types of a certain severity: Fig. 11 summarises the results acquired from this test case. The bar plot presents the mean AUROC values averaged over all test noise types when FACER is trained on all noise types of a specific severity level. This shows that training on the lowest severity level of noise type is most beneficial.

d) Detecting untrained dataset samples: We employed two untrained datasets, JAAD test set with random boxes added on the data and Cityscapes test data in this test case. Table II shows the AUROC values. From the AUROC value of 1.0, we can deduce that FACER is able to recognise the random boxes on JAAD data as anomalies. The high AUROC values yielded from Cityscapes data suggest that FACER performs well on unseen (out-of-distribution) data.

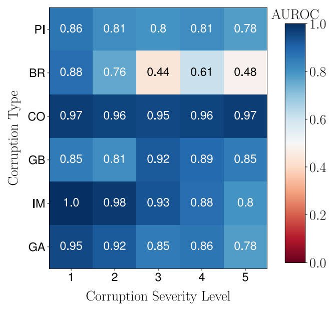


Fig. 9. FACER AUROC obtained with noise not trained on.

Fig. 10. Examples of JAAD data with varied brightness (images taken from the training dataset).

V. Conclusion and Outlook

Analysing the PDET performance on data sequences exhibits unexpected, reproducible errors. By doing so we leverage our knowledge on the PDET reliability as well as on the data. This enables effective training data optimisation and further measures to increase reliability. Therefore, we emphasise the importance of both DNN and data for ensuring reliable PDET behaviour.

Since not all suitable data can be determined during development, we cannot ensure robust behaviour in all situations at run-time. We evaluate this by identifying augmented data that cause unreliable PDET behaviour. In consequence, we complement data suitability and propose FACER, a run-time anomaly detector. Similar to training data optimisation, FACER also considers the behaviour of the DNN. This distinguishes it from other anomaly detectors taking only data as an input.

Further elaboration on merging the outputs of FACER and PDET might contribute to improving the overall safety of an automated vehicle.

TABLE II
AUROC values obtained using FACER on untrained data being trained with JAAD data

Test Data	height x width	AUROC
JAAD with random erasing (grey box)	1080 x 1920	1.00
Cityscapes (original)	1024 x 2048	0.90
Cityscapes (cropped)	1080 x 1920	0.90

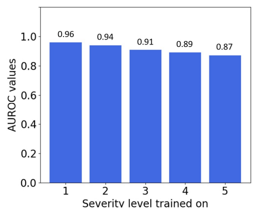


Fig. 11. Mean AUROC values obtained using FACER with all noise types for each severity level.

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, vol. 28, pp. 91–99.
[2] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, pp. 1–48, 2019.
[3] U. Franke, "30 years fighting for robustness," June 2018, talk at Robust Vision Challenge 2018. [Online]. Available: https://www.youtube.com/watch?v=AOP8iitFbPY
[4] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *ICLR*, 2019.
[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, vol. 27, pp. 2672–2680.
[6] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," 2017. [Online]. Available: http://arxiv.org/abs/1710.10196
[7] L. Gauerhof and N. Gu, "Reverse variational autoencoder for visual attribute manipulation and anomaly detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 2103–2112.
[8] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W. Ku, and A. Nguyen, "Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects," *CoRR*, vol. abs/1811.11553, 2018. [Online]. Available: http://arxiv.org/abs/1811.11553
[9] H. Kato, Y. Ushiku, and T. Harada, "Neural 3d mesh renderer," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3907–3916.
[10] L. Gauerhof, R. Hawkins, C. Picardi, C. Paterson, Y. Hagiwara, and I. Habli, "Assuring the safety of machine learning for pedestrian detection at crossings," in *Computer Safety, Reliability, and Security*. Springer, 2020.
[11] F. N. Iandola, S. Han, M. W. Moskewicz, A. K, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and $<$ 0.5MB model size," 2016.
[12] I. Kotseruba, A. Rasouli, and J. K. Tsotsos, "Joint attention in autonomous driving (JAAD)," 2016.
[13] A. Rasouli, I. Kotseruba, and J. K. Tsotsos, "Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 206–213.
[14] ——, "Its not all about size: On the role of data properties in pedestrian detection," in *ECCV Workshops*, 2018.
[15] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *AAAI*, 2020, pp. 13 001–13 008.
[16] C. Schorn and L. Gauerhof, "FACER: A universal framework for detecting anomalous operation of deep neural networks," in *Proceedings of the IEEE Inteligent Transportation Systems Conference*, 2020.
[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

FACER: A Universal Framework for Detecting Anomalous Operation of Deep Neural Networks

Christoph Schorn and Lydia Gauerhof
Bosch Research, Renningen, Germany
Email: firstname.surname@de.bosch.com
The authors contributed equally to this work.

***Abstract*—The detection of anomalies during the operation of deep neural networks (DNNs) is of essential importance in safety-critical applications, such as autonomous vehicles. In the field, classifiers may face rare environmental conditions, unknown objects, hardware failures, and other types of anomalies. Nevertheless, DNNs still predict arbitrarily high class probabilities in these cases and are unable to recognize out-of-distribution operation modes. In this paper, we introduce FACER, an efficient and versatile framework that is trainable to detect various types of anomalies in pre-trained DNNs. FACER operates on compressed intermediate feature representations of the supervised network that can be easily obtained. We evaluate the detection of different input corruptions as well as outliers drawn from out-of-distribution datasets with CIFAR-10, CIFAR-100 and SVHN classification models. The detection performance of our method is on par with other state-of-the-art methods, while our method can be easier implemented and integrated into resource-constrained hardware systems.**

***Index Terms*—deep neural networks, out-of-distribution detection, anomaly detection, sensor noise, robust classification**

I. Introduction

For safety-critical systems, such as automated vehicles, correct and precise sensing is prerequisite for safe behavior. Therefore, oversight of an object or erroneous object detection has to be avoided. Otherwise trajectory planing and decision making will be based on incomplete or wrong environmental information and might cause hazardous situations.

There are different kinds of anomalies that can lead to incorrect reasoning in machine learning algorithms. Especially, deep neural network (DNN) based classification performs best on data that were seen during training. Input data not belonging to the training distribution are regarded as out-of-distribution (OOD). Special weather conditions, such as rainfall, and lighting conditions, affect the quality of input data. Furthermore, sensors and other components of data processing, such as filters, manipulate data. Some of these variations might occur seldom or not at all in training data and can consequently result in OOD conditions.

Our approach for anomaly detection is based on a trainable **f**eature **a**ctivation **c**onsistency check**er** (FACER), receiving traces from the intermediate outputs of a main DNN as input. Fig. 1 depicts our approach. Anomalies in the input lead to inconsistencies in the feature representation which can be detected by FACER. Previous work by Schorn et al. [30] has shown that this approach can be used to detect inconsistencies

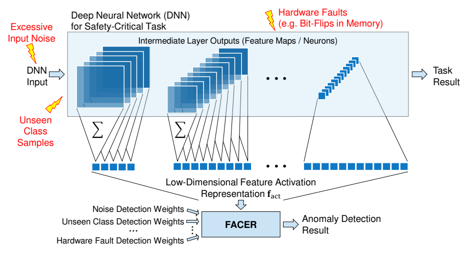

Excessive Input Noise
Deep Neural Network (DNN) for Safety-Critical Task
Hardware Faults (e.g. Bit-Flips in Memory)
Intermediate Layer Outputs (Feature Maps / Neurons)
DNN Input
Task Result
Unseen Class Samples
Low-Dimensional Feature Activation Representation f_{act}
Noise Detection Weights
Unseen Class Detection Weights
Hardware Fault Detection Weights
FACER
Anomaly Detection Result

Fig. 1. **F**eature **a**ctivation **c**onsistency check**er** (FACER) can be trained to detect various types of anomalies in DNNs.

in the computing hardware, such as bit-flips. In this paper, we extend the analysis to OOD detection.

Note that FACER differs from anomaly detection methods operating on the input samples, since it exploits the feature extraction capabilities of the main network. Thus, FACER can use a small classifier for anomaly detection. Furthermore, FACER is applicable to existing DNNs without the need to retrain them. Only the anomaly detector itself has to be trained and for this only binary anomaly or non-anomaly labels but no other labeling information is required. This results in reduced training effort and increased flexibility during development.

Moreover, FACER can be applied to the same feature activation input with different sets of trainable weights. This enables an efficient check for multiple kinds of anomalies.

In this paper, we make the following contributions:

- We introduce an efficient and effective approach for the concurrent detection of multiple anomalies during the operation of DNNs.
- We evaluate detection performance and generalization abilities on eight noise types with different severities applied to images of the CIFAR-10, CIFAR-100, and SVHN tasks. We find that training on low severities makes the anomaly detector generalize well to higher severity levels. Furthermore, by combining multiple noise types during training, we achieve high detection capabilities also on noise types not seen during training.

- We also evaluate the detection of unseen class samples. The detection performance of our method is in the range of other state-of-the-art methods for this task, while it offers the advantage of being more flexibly applicable to other anomaly detection problems as well.

II. Related work

Before introducing our approach in detail, we give an overview on related work. Firstly, we review different types of anomalies, and secondly, we discuss existing measures against anomalies.

A. Anomalies

"Anomalies are patterns in data that do not conform to a well defined notion of normal behavior." [6]. In the following, anomalies regarding hardware operation, adversarial attacks, and out-of-distribution data are introduced.

Hardware Failure. Hardware is affected by reliability threats [11]. For example, high energy particle strikes can result in bit-flip errors [4]. Moreover, aging effects result in degradation of a circuit's characteristics and may lead to errors in computation [11]. Since computing errors can lead to silent data corruption, it is desirable to detect them using anomaly detection methods [30].

Adversarial Attacks. Other fields of anomalies can be associated with security threats [11]. Attackers can corrupt data and computation. One example are adversarial attacks that fool a neural network by adding small distortions to the input data [5], [21]. Novel defenses and approaches for robustness [24], [26], [34] are followed by novel attacks [1], [7]. Physical attacks also play a role [9]. Adversarial attacks can be considered as a type of worst-case robustness analysis for DNNs [12].

Out-Of-Distribution. Performance evaluation of machine learning is based on the assumption that training and test data are sampled from similar distributions. DNNs tend to fail when data distributions during training and operation differ from each other [13]. Anomalies are regarded as outliers of the nominal distribution given by the training set. However, anomalous data points are not limited to the tails of the nominal distribution [8]. For computer vision tasks perturbations and corruption of data are often found due to different weather and lighting conditions as well as signal processing effects. Hendrycks and Dietterich provide a benchmark for common image corruptions and perturbations [12]. We treat such anomalies as OOD, based on the assumption that they are not included in the training data set. Moreover, data including unseen objects or increased environmental scope are also regarded as OOD [13].

In this paper, we focus on out-of-distribution anomalies, because we consider them as the most commonly occurring threat in safety-critical DNN applications.

B. Measures Against Anomalies

DNNs with a softmax classifier are known to be overconfident even for anomalies [2], [10]. To overcome this drawback, there are different approaches, such as increasing robustness, uncertainty measures, and anomaly detection.

Robustness and Uncertainty. The robustness against anomalies can be increased by training, if anomalies are known or occur frequently. Nevertheless, input distribution might change over time and different anomalies will occur. For this reason, robustness against unknown anomalies cannot be guaranteed. In this regard, well-calibrated predictive uncertainty is indispensable for real-word applications [18], [22], [28]. However, many uncertainty approaches cannot be applied in safety-critical real-time applications, due to their high computational demands and latency. We thus focus on efficient anomaly detection methods instead.

Anomaly Detection. Anomaly detection methods can be categorized into input-based and inference-based. We follow an inference-based approach in this paper. *Input-based* anomaly detectors often utilize generative models, especially variational autoencoders (VAEs) and generative adversarial networks (GANs) (e. g. [3]). Meinke and Hein [25] recently proposed an input-based anomaly detection approach that achieves state-of-the-art performance. However, their method requires a joint training of the DNN classifier for the actual task and density estimators for in- and out-distribution, which results in reduced flexibility compared to our approach. *Inference-based* models, on the other hand, operate on the output or feature space of DNNs. Lee et al. [22] proposed a method to measure the probability density of test samples within feature spaces to detect OOD or adversarial samples. They apply the concept of a generative (distance-based) classifier to any pre-trained softmax neural classifier without re-training in such a way that the Mahalanobis distance is used for the confidence score. Liang et al. [23] proposed the method ODIN that uses temperature scaling after adding small perturbations to the input in order to separate the softmax score distribution between in- and out-of-distribution images. Hendrycks et al. [14] proposed outlier exposure (OE), which achieves state-of-the-art OOD detection performance. OE fine-tunes the classifier of the actual task to enforce uniform probability predictions on a large out-distribution set. Nevertheless, retraining the main classifier may not be desirable in some cases. Metzen et al. [26] introduced a method that, like ours, works on intermediate feature representations for the task of adversarial example detection. However, their choice of feature representation requires more manual fine-tuning than our approach.

III. FACER: Feature Activation Consistency checkER

In the following, we introduce a universal framework for anomaly detection in DNNs. We call this method **f**eature **a**ctivation **c**onsistency check**er** (FACER). FACER builds up on previous work presented by Schorn et al. [30].

An overview of FACER's basic principle is depicted in Fig. 1. Based on the supervised DNN's intermediate outputs, a feature activation vector $\mathbf{f}_{\text{act}}$ is computed. FACER then performs a binary classification into anomaly and non-anomaly based on $\mathbf{f}_{\text{act}}$ and a set of learned weights for the respective anomaly type. FACER can be trained on detecting various types of anomalies, such as random bit-flips in the

DNN computing hardware [30], as well as OOD samples with noise or unseen classes. This makes it universally applicable as a consistency checker for monitoring safety-critical neural network applications. Moreover, $\mathbf{f}_{\mathrm{act}}$ has to be computed and transferred to FACER only a single time to perform multiple anomaly checks, which makes this method suitable for resource-constrained applications.

A. *Computing Feature Activations*

As shown in Fig. 1, the input to FACER is a vector $\mathbf{f}_{\mathrm{act}}$ in which the neuron outputs of all layers of the DNN are concatenated. These outputs are generated from a single input sample given to the DNN. In the case of 2D convolutional layers, which are commonly used in image classification DNNs, the layer output consists of multiple 2D feature maps corresponding to the different filter kernels of the layer. For each feature map we append a single value to $\mathbf{f}_{\mathrm{act}}$ by summation over all values of the feature map.

The benefit of accumulation over feature maps is twofold. Firstly, it adds a degree of invariance against input image transformations such as rotation and zoom to the feature activation representation. This is desirable, since those transformations happen naturally in mobile vision applications and we therefore do not consider them as OOD. Secondly, accumulation results in a comparatively low-dimensional feature activation representation. This reduces the required parameters and operations of FACER. Furthermore, data transfer from the task DNN to FACER is reduced, which is especially useful for system architectures in which FACER is implemented on separated safety-monitoring hardware.

We normalize each $\mathbf{f}_{\mathrm{act}}$ by subtracting mean and dividing by standard deviation (std) values per element obtained over the non-anomaly $\mathbf{f}_{\mathrm{act}}$ training set. To avoid division by zero, we add a small offset of 10^{-8} to the standard deviation.

An example of resulting mean and std feature values computed over anomaly (CIFAR-10) and non-anomaly (CIFAR-100) test sets is shown in Fig. 2. Normalization parameters were derived from the CIFAR-100 training set. Note that the mean values of anomaly data deviate stronger from zero than those of non-anomaly data, but the deviations are small in comparison to the standard deviation. Thus, a simple linear classifier would not be sufficient to achieve a good separation of anomaly and non-anomaly data. FACER outperforms state-of-the-art methods on this specific task example by a large margin, as will be shown in Section IV-C.

B. *Training the FACER Classifier*

While FACER can in principle use any trainable binary classifier, we decided to utilize a small feedforward neural network that takes $\mathbf{f}_{\mathrm{act}}$ as input and classifies it into anomaly or non-anomaly. An architecture with two hidden layers, each with 64 neurons and rectified linear unit (ReLU) activation functions, and an output layer with a single sigmoid neuron has proven to work well for our purposes. We employ standard supervised training techniques using binary cross-entropy loss, backpropagation, and the ADAM [19] optimizer with default hyperparameters as recommended by its authors.

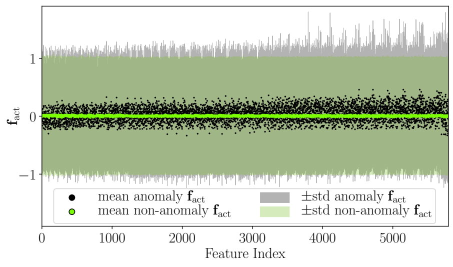


Fig. 2. Mean and std of normalized $\mathbf{f}_{\mathrm{act}}$ per feature computed over anomaly and non-anomaly test sets. In this example the DNN is trained on CIFAR-100 data (non-anomalies) and anomalies correspond to CIFAR-10 test samples.

We chose a batch size of 4096. Moreover, we apply batch normalization [17] before ReLU activations during training.

A training dataset is required to learn anomaly detection. In this paper we focus on OOD detection. Thus, training data are created by presenting in-distribution samples from the distribution $\mathcal{D}_{\mathrm{in}}^{\mathrm{train}}$ on which the DNN regularly operates as well as OOD samples from a distribution $\mathcal{D}_{\mathrm{out}}^{\mathrm{train}}$. Typically not all possible types of OOD data are known at design time. Therefore, $\mathcal{D}_{\mathrm{out}}^{\mathrm{train}}$ should be chosen to make FACER generalize well across different types of OOD data. This will be further examined in the experimental section of this paper.

For each sample presented to the DNN, the feature activations $\mathbf{f}_{\mathrm{act}}$ are recorded and labeled with a binary label classifying it as anomalous or non-anomalous. Test data for the evaluation of FACER are created similarly to training data but using DNN input samples from a distinct dataset with distribution $\mathcal{D}_{\mathrm{out}}^{\mathrm{test}}$. Throughout our experiments we create balanced datasets, which means that the number of anomaly and non-anomaly samples in the dataset is equal. This is beneficial for training as well as for a meaningful evaluation.

IV. Experiments

We evaluate FACER on two different OOD detection problems: (1) detection of excessive noise in the input data; (2) detection of unseen classes. Performance on bit-flip error detection has been previously evaluated by Schorn et al. [30].

A. *Preliminary Remarks*

We use area under the receiver operating characteristic (AUROC) as metric for evaluating the anomaly detection performance of FACER. AUROC corresponds to the probability that a randomly chosen negative (non-anomaly) sample will have a smaller estimated probability of belonging to the positive (anomaly) class than a randomly chosen positive example [16]. We run every experiment 10 times, randomly shuffling the training data, and report mean AUROC values.

Throughout our experiments we use three image classification DNNs trained on the CIFAR-10 [20], CIFAR-100 [20], and SVHN [27] tasks respectively. The DNNs are based on the DenseNet [15] architecture with a depth of 40 and growth rate of 12. They have approximately 600k trainable parameters and achieve classification accuracies of 94,39% (CIFAR-10), 75,10% (CIFAR-100), and 96,95% (SVHN). The resulting dimension of $\mathbf{f}_{\text{act}}$ is 5704 for the CIFAR-10 and SVHN classifiers, and 5794 for the CIFAR-100 classifier.

To increase the detection performance of FACER, we enlarge its training sets by applying random transformations (shift, mirroring, rotation, zoom, and color channel shift) to images from which $\mathbf{f}_{\text{act}}$ are generated. Such transformations are commonly used when training classifiers [29].

B. Evaluation of Noise Detection

The presence of input noise that has not been observed in the training data distribution can severely reduce the classification accuracy of DNNs [12]. For safety-critical applications, such as autonomous vehicles, it is essential to recognize such situations in order to activate appropriate safety measures. To provide comparability, we evaluate the noise detection capability of FACER on widely used image classification benchmarks, though FACER can also be applied to other data types, such as radar or sound signals.

Similar to [12], we consider eight different noise types, as shown in Fig. 3. *Gaussian noise (GA)* is additive and independent at each pixel. It occurs during image acquisition due to the inherent electronic circuit noise of the sensor. *Shot noise (SH)* arises from the discrete nature of photons and follows a Poisson distribution. *Impulse noise (IM)* can result from bit errors during transmission or in the sensor's analog-to-digital converters. *Speckle (SP)* results from interference and is present in images obtained from coherent imaging systems, such as synthetic aperture radar, laser, and ultrasound. *Gaussian blur (GB)* averages the value of each pixel with its neighborhood by convolving the image with a Gaussian function. It mimics the effect caused e. g. by image acquisition through fog. *Contrast (CO)* can be low in difficult lighting conditions, e. g. at night or during cloudy weather. *Pixelation (PI)* is caused when an image is upscaled to a higher resolution. *Brightness (BR)* corrupts an image when the pixel readout values saturate at high light intensities.

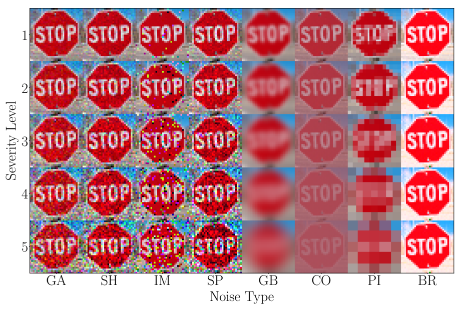


Fig. 3. Noise types evaluated in this paper. Severity levels are defined based on multi-scale structural similarity (MS-SSIM).

We use five different noise severity levels. The severity levels $1, 2, \ldots, 5$ correspond to an average reduction of the multi-scale structural similarity (MS-SSIM) [33] to $0.9, 0.8, \ldots, 0.5$, respectively, where MS-SSIM is measured between a set of original images and their noisy variants. This approach was also used by Hendrycks et al. [12], as confirmed on request.

In the following we evaluate three different scenarios of training FACER for excessive input noise detection. In these scenarios the anomalous sample distributions for training $\mathcal{D}_{\text{out}}^{\text{train}}$ correspond to: (1) a single noise type and severity; (2) all noise types of a certain severity; (3) all except one noise types of a certain, while the test distribution.

Training on single noise type. The resulting AUROC values for all possible combinations of single noise type training and testing for the CIFAR-100 dataset are shown in Fig. 4. Results for the other two datasets are similar.

It can be seen that when training and test anomaly distribution are the same (values along the diagonal line from bottom left to top right), FACER delivers almost perfect classification performance. Recall that an AUROC of 1 is best and an AUROC of 0.5 corresponds to random guessing. Furthermore, generalization between different severity levels of the same noise type (5×5 blocks on the diagonal from bottom left to top right) is generally quite high, while especially training on a low severity level lets FACER also detect higher severity levels. However, the opposite is not always true, as can be observed for instance in the case of contrast (CO) noise.

Generalization between different noise types works well in several but not all cases. Especially for Gaussian, shot, impulse and speckle noise, training FACER on one of them lets it also detect the others correctly. This can be explained by the similar effect these noise types have on the image, namely the addition of high frequency components. On the

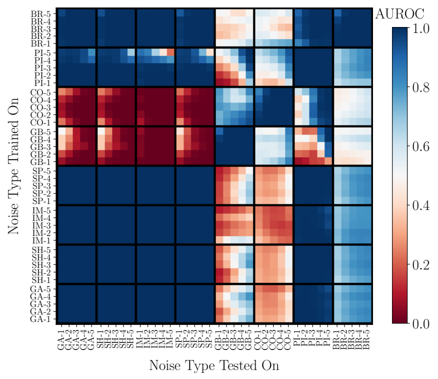


Fig. 4. Resulting AUROC values (average of 10 runs) on CIFAR-100 task for all possible combinations of noise types applied during training and test of FACER. Two letter acronyms correspond to the noise types described before and numbers indicate severity levels. The plot is best viewed in color.

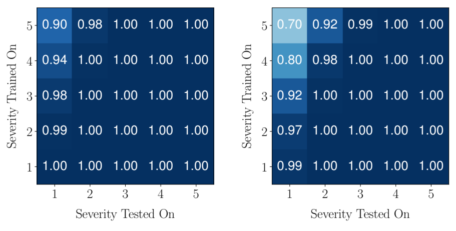
TABLE I
RESULTING AUROC VALUES (AVERAGE OF 10 RUNS) AFTER TRAINING FACER ON MULTIPLE NOISE TYPES WITH A SEVERITY LEVEL OF ONE AND EVALUATING IT ON INDIVIDUAL NOISE TYPES WITH ALL FIVE SEVERITY LEVELS

	Trained Noise $\mathcal{D}_{\text{out}}^{\text{train}}$	All Types								All Except Tested Type							
	Tested Noise $\mathcal{D}_{\text{out}}^{\text{test}}$	GA	SH	IM	SP	GB	CO	PI	BR	GA	SH	IM	SP	GB	CO	PI	BR
Model	CIFAR-10	1.000	1.000	1.000	1.000	1.000	0.998	1.000	0.999	1.000	1.000	1.000	1.000	0.910	0.851	0.955	0.730
	CIFAR-100	1.000	1.000	1.000	1.000	1.000	0.997	1.000	0.998	1.000	1.000	1.000	1.000	0.835	0.805	0.974	0.652
	SVHN	1.000	1.000	1.000	1.000	0.996	0.990	1.000	0.997	1.000	1.000	1.000	0.999	0.866	0.875	0.946	0.640

other hand, Gaussian blur and contrast noise act as low-pass filters, i. e. they remove high frequencies from the image. This explains why training on them results in low AUROC values when testing on high-frequency noise types. In fact, most AUROC values are close to zero for these cases, which means that the classifier can distinguish the anomaly and non-anomaly classes quite well, but does this the wrong way round, because it was trained on the opposite type of noise. Pixelation has a mixture of low-frequency and high-frequency behavior. High-frequencies result from the vertical and horizontal edges that are added by pixelation. This is more pronounced at low severities and explains bad test performance on Gaussian blur and contrast noise in these cases. On the other hand, at high pixelation severity, the averaging (i. e. low-pass) effect of pixelation prevails and AUROC in high-frequency noise test cases slightly drops. Best overall generalization is seen when training on brightness noise.

Training on all noise types. Since having a separate anomaly detection model for each noise type results in a large number of detection runs that have to be performed, we now evaluate the ability of FACER to be trained on multiple noise types simultaneously. In a preliminary evaluation, we decide which noise severity level should be used during training. Fig. 5 shows for the CIFAR-100 benchmark that applying the lowest severity level during training is most beneficial, since close to one AUROC is achieved for all test severities in this case. Thus, in the following, we always use severity level one during training, while testing over all severity levels.

AUROCs for individual test noise types with a training on all noise types are listed in the left part of TABLE I. Since all values are at least 0.99, we conclude that training a single



(a) Mean AUROC (b) Minimum AUROC

Fig. 5. Resulting mean and minimum AUROC values on the CIFAR-100 task when FACER is trained on all noise types with a certain severity level and evaluated on all noise types with a certain severity level.

FACER model for the task of detecting multiple different noise types works very well.

Training on all except one noise type. In practice, it cannot be assumed that all noise types are known at design time of the classifier. Thus, we perform a further experiment, where we exclude one of the noise types at a time from the training set and test detection performance on this excluded noise type. The results are shown in the right part of TABLE I. As in the single noise training case, generalization across high-frequency noise types (GA, SH, IM, SP) works well. Gaussian blur, contrast, and pixelation can also be detected to a considerable extent. However, brightness noise turns out to be most difficult to detect, when it is not included in the anomaly training set.

C. Evaluation of Unseen Classes Detection

Having evaluated the noise detection capabilities of FACER, we now take a look at the task of detecting samples with classes that were not observed during training. This task is more commonly considered in the literature and we benchmark the performance of FACER against state-of-the-art approaches OE [14] and certified certain uncertainty (CCU) [25].

A proper training set with unseen class samples is required to generate anomalous $\mathbf{f}_{\text{act}}$ values for training FACER. State-of-the-art methods, [14], [25], employ the 80 Million Tiny Images dataset [32] as OOD training data, because of its large size and diversity. We noticed that FACER manages to achieve good AUROC performance with a much smaller training set, namely Tiny-ImageNet [31]. This dataset has only 100 000 training images and 200 classes.

TABLE II reports the resulting AUROC values of our experiments with FACER and values for competing methods taken from [25]. For each of the three image classifiers, we take its test data as in-distribution samples $\mathcal{D}_{\text{in}}^{\text{test}}$ and evaluate

TABLE II
RESULTING AUROC VALUES (AVERAGE OF 10 RUNS) OF FACER ON UNSEEN CLASS OOD DETECTION IN COMPARISON TO STATE-OF-THE ART METHODS (OE AND CCU VALUES TAKEN FROM [25])

$\mathcal{D}_{\text{in}}$	$\mathcal{D}_{\text{out}}^{\text{test}}$	OE [14]	CCU [25]	FACER
CIFAR-10	CIFAR-100	0.953	0.942	**0.966**
	SVHN	**0.988**	0.982	0.985
CIFAR-100	CIFAR-10	0.816	0.802	**0.965**
	SVHN	0.935	**0.942**	0.918
SVHN	CIFAR-10	**1.000**	**1.000**	**1.000**
	CIFAR-100	**1.000**	**1.000**	**1.000**

test images of the respective other two datasets as OOD distributions $\mathcal{D}_{\text{out}}^{\text{test}}$. We see that FACER delivers state-of-the-art OOD detection performance in four out of six cases, and is close to the reference methods in the other two cases.

V. Conclusions

With FACER we propose a versatile and efficient framework for detecting multiple types of anomalous DNN operation modes. Such anomaly detection is of essential importance for DNN-based perception in safety-critical systems, for instance autonomous vehicles. Our method achieves state-of-the-art out-of-distribution detection performance.

FACER utilizes the principle of consistency checking across intermediate feature representations of a DNN. It can be applied to existing pre-trained classifiers without the need to retrain them. Moreover, FACER uses machine learning to train a comparatively small detector on different types of anomalies. Thus, it can be flexibly extended with further anomaly detection capabilities. Since intermediate feature representations have to be computed and transmitted only once in order to test for multiple anomalous operation modes, FACER is efficient in cases where a full safeguarding against hardware failures, sensor noise, distributional shift, and other types of anomalies is required.

Our experiments have shown that FACER is able to generalize well between multiple types of input corruptions. In the case of unseen classes detection, FACER requires a much smaller out-distribution training set than existing methods to achieve state-of-the-art performance.

References

[1] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W. Ku, and A. Nguyen. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4849, 2019.

[2] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in AI safety. *arXiv e-prints*, 2016.

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[4] R. C. Baumann. Radiation-induced soft errors in advanced semiconductor technologies. *IEEE Transactions on Device and Materials Reliability*, 5(3):305–316, 2005.

[5] N. Carlini, G. Katz, C. Barrett, and D. L. Dill. Provably minimally-distorted adversarial examples. *arXiv e-prints*, 2017.

[6] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 2009.

[7] Y. Dong, Q.-A. Fu, X. Yang, T. Pang, H. Su, Z. Xiao, and J. Zhu. Benchmarking adversarial robustness. *arXiv e-prints*, 2019.

[8] A. F. Emmott, S. Das, T. Dietterich, A. Fern, and W.-K. Wong. Systematic construction of anomaly detection benchmarks from real data. In *ACM SIGKDD Workshop on Outlier Detection and Description*, ODD 13, pages 16–21, 2013.

[9] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust physical-world attacks on deep learning visual classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference for Learning Representations*, 2015.

[11] M. A. Hanif, F. Khalid, R. V. W. Putra, S. Rehman, and M. Shafique. Robust machine learning systems: Reliability and security for deep neural networks. In *IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, pages 257–260, 2018.

[12] D. Hendrycks and T. G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference for Learning Representations*, 2019.

[13] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference for Learning Representations*, 2017.

[14] D. Hendrycks, M. Mazeika, and T. Dietterich. Deep Anomaly Detection with Outlier Exposure. In *International Conference for Learning Representations*, 2019.

[15] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2261–2269, 2017.

[16] J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3):299310, 2005.

[17] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015.

[18] A. Kendall and Y. Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances in Neural Information Processing Systems 30*. 2017.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference for Learning Representations*, 2015.

[20] A. Krizhevsky. Learning multiple layers of features from tiny images. Master Thesis, University of Toronto, 2009.

[21] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *Workshop Track of the International Conference for Learning Representations*, 2017.

[22] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7167–7177. Curran Associates, Inc., 2018.

[23] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*, 2018.

[24] J. Lu, H. Sibai, E. Fabry, and D. Forsyth. Standard detectors aren't (currently) fooled by physical adversarial stop signs. *arXiv e-prints*, 2017.

[25] A. Meinke and M. Hein. Towards neural networks that provably know when they don't know. *arXiv e-prints*, 2019.

[26] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations*, 2017.

[27] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[28] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek. Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift, 2019.

[29] L. Perez and J. Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv e-prints*, 2017.

[30] C. Schorn, A. Guntoro, and G. Ascheid. Efficient on-line error detection and mitigation for deep neural network accelerators. In B. Gallina, A. Skavhaug, and F. Bitsch, editors, *Computer Safety, Reliability, and Security (SAFECOMP)*, volume 11093 of *LNCS*. Springer, 2018.

[31] Stanford CS231n. Tiny ImageNet Visual Recognition Challenge. [Online] Available: https://tiny-imagenet.herokuapp.com/.

[32] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

[33] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *37th Asilomar Conference on Signals, Systems Computers*, volume 2, pages 1398–1402, 2003.

[34] E. Wong, F. Schmidt, J. H. Metzen, and J. Z. Kolter. Scaling provable adversarial defenses. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8400–8409. Curran Associates, Inc., 2018.

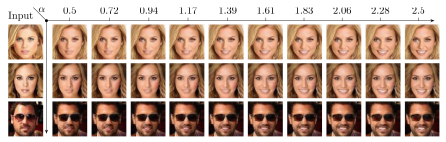
Reverse Variational Autoencoder for Visual Attribute Manipulation and Anomaly Detection

Lydia Gauerhof*
Corporate Research, Robert Bosch GmbH
lydia.gauerhof@de.bosch.com

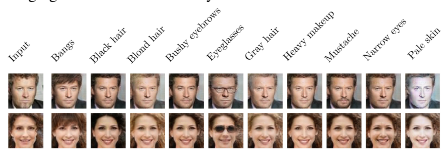
Nianlong Gu*
Institute of Neuroinformatics, ETH Zurich
niangu@ethz.ch

Abstract

In this paper, we introduce the 'Reverse Variational Autoencoder" (Reverse-VAE) which is a generative network. On the one hand, visual attributes can be manipulated and combined while generating images. On the other hand, anomalies, meaning deviations from the data space used for training, can be detected. During training the generator network maps samples from stochastic latent vectors to the data space. Meanwhile the encoder network takes these generated images to reconstruct the latent vector. The generator and discriminator are trained adversarially. The discriminator is trained to distinguish between real and generated data. Overall, our model tries to match the joint latent/data-space distribution of the generator and the latent/data-space joint distribution of the encoder by minimizing their Kullback-Leibler divergence. Desired visual attributes of CelebA images are successfully manipulated. The performance of anomaly detection is competitive with state-of-the-art on MNIST.



(a) The first column is original serious faces. From second to last column, the dominance of a smile increases with the scale factor α changing from 0.5 to 2.5 linearly.



(b) Original images in first column are manipulated.

Figure 1: Adding single visual attributes

1. Introduction

Highly automated driving (HAD) has the potential to revolutionize the way we travel. At the same time, HAD is a safety-critical application in which the violation of safety goals, e.g. a crash with other road users is not acceptable. Consequently, when used for HAD, DL models such as Deep Neural Networks (DNNs) are required to perform robustly [9, 15, 2, 14, 11] despite all kinds of anomalies. Causes for anomalies include lack of diversity in training data set, or changes in sensors over time which may result in shift of distribution of captured images with respect to training data set [16, 29]. Therefore, it is important to detect anomalies - whether the current input image of a DNN is beyond the feature distribution of the training images data set [32, 5]. Then these anomalies can be included to the data set, e.g. by data augmentation based on attribute manipulation, in order to increase robustness.

Images can be regarded as high dimensional vectors which is challenging to analyze the distribution directly. Luckily, the fact that images usually have patterns like human faces indicates that the distribution of a set of images may lie in a low dimensional manifold. This has been experimentally proven by the success of generative adversarial models [12, 4, 18] which can generate various and realistic images. However, a GAN [12] type structure can only learn the mapping from low dimensional latent space to the images space. It is still challenging to both get the embedding of an image and generate new images in a decent way.

Variational Autoencoder (VAE) is one of the earliest model which aims to do both image encoding and image generation. Although VAE can learn meaningful image embedding that can be used for data distribution analysis, it tends to generate images with blurring effects which limits its usage in image generation and manipulation tasks. Inspired by VAE, more advanced models like VAE-GAN

*equal contribution

[22] and ALI [8] were proposed with the goal of improving the image generation performance while keeping the ability of encoding input images to latent space. These models involve in either picking up a certain hidden layer of the discriminator as feature-wise representation, or adopting a sophisticated model structure. Moreover, in these works the experiments are mainly done on 64×64. It remains unclear if those models can be well scaled to deal with images with larger resolution in a more practical scenario.

In this work, we introduce the "Reverse Variational Autoencoder" (Reverse-VAE) which can not only learn an accurate mapping to low dimensional space, but also generate realistic and diverse images. Moreover, our model is compatible with the recently proposed progressively growing strategy [18] to process high resolution images with good scalability. Our model can be reconfigured such that it is used either for anomaly detection or for visual attribute manipulation as a data augmentation method to improve the DNN model robustness against anomalies [1, 10, 28, 34, 33] (examples are found in fig. 1). Our contributions are:

- **A novel form of training settings** reduces the gap between joint latent/data distribution of generator and the joint distribution of encoder by minimizing the Kullback-Leibler divergence. Image **generation / reconstruction** are competitive with state-of-the-art.
- A **simple architecture** makes our model **easy to train with less parameter tuning** and able to be **up-scaled to generate and reconstruct high resolution images** using a PGGAN [18] setting.
- Good reconstruction performance is **restricted on distribution of training data** enables the model to perform well in detecting anomalies.
- For manipulating visual attributes the model is trained **without auxiliary information**, such as labeled attributes. After training we extract **dedicated visual attribute vectors in the latent space** using a small subset of labeled images. We gain flexibility in manipulating new attributes without retraining the model.
- **Combining both applications** leads to a reduced training effort and to an increased development efficiency.

Although in this paper the attribute manipulation and detected anomalies do not necessarily rely on each other, this approach strengthens the development of a unified model for detecting an anomalies and extracting the according attributes in order to augment data.

2. Related Work

First, deep generative models, such as Generative Adversarial Network (GAN) [12] and Variational Autoencoder (VAE) [20], modeling high dimensional data sets are explained. Second, models combining aspects of VAE and GAN are introduced and difference to our model are discussed.

2.1. GAN, VAE and their extensions

The GAN [12] generates more realistic images by making use of an adversarial training procedure. A discriminator learns to distinguish the real images from the images synthesized by a generator. At the same time, the generator tries to "fool" the discriminator by generating more realistic images. Wasserstein-GAN (WGAN) solves the gradient vanishing and mode collapse problem of the original GAN [4] with a minmax game of the Wasserstein distance. Moreover, Chen *et al.* [6] proposed an information-theoretic extension to the GAN (InfoGAN) which is able to learn disentangled representation of limited visual attributes, such as the rotation or stroke of MNIST [23] digits. Nevertheless, the GAN-type models cannot learn a low dimensional embedding as we need for feature distribution analysis.

The VAE predicts the posterior distribution over the latent variables by employing an encoder, and uses an decoder to reconstruct the images given the encoder output [20]. These generated images usually look blurred, though. The Conditional Variational Autoencoder (CVAE) and its variants are proposed for structured output prediction based on the conditional deep generative model with known label information [35]. CVAE is not suitable for our purpose, since we want a model to disentangle information without given auxiliaries during training.

2.2. Models combining aspects of VAE and GAN

After VAE and GAN were proposed, models combining different aspects of VAE and GAN have evolved: for example Adversarial Autoencoder (AAE), VAE-GAN and Adversarially Learned Inference (ALI).

The AAE is a probabilistic autoencoder including a GAN to conduct variational inference by meeting the aggregated posterior of the latent vector with an arbitrary prior distribution [26]. Compared with the original images, the generated images still look blurred.

Apart from AAE, there is also VAE-GAN combining VAE with GAN such that the learned feature representations in the GAN's discriminator are used as a basis for the VAE reconstruction loss [22]. In VAE-GAN the feature-wise reconstruction loss is define as

$$L_{\text{recon\_x}} = \|\text{Dis}_l(\boldsymbol{x}) - \text{Dis}_l(\hat{\boldsymbol{x}})\|_2^2 \qquad (1)$$

where $\text{Dis}_l(\boldsymbol{x})$ means the l^{th} hidden layer of the discriminator, and $\boldsymbol{x}$, $\hat{\boldsymbol{x}}$ are input images and reconstructed images respectively. Our model differentiates from VAE-GAN in a way that we did not use such a feature-wise reconstruction loss primarily, with less parameter tuning such as the selection of l over different data set or different model size. Experiments show that our model tends to balance the tasks of generating high quality images and accurately reconstructing the input images more properly.

Furthermore, the model Adversarially Learned Inference (ALI) was proposed to learn a generation network (generator) and an inference network (encoder) using an adversarial framework [8]. A discriminator is trained to distinguish between joint samples $(\tilde{\boldsymbol{z}}, \boldsymbol{x})$ of the data and the corresponding latent vector from the encoder and the joint samples $(\boldsymbol{z}, \tilde{\boldsymbol{x}})$ from the generator. At the same time the encoder and generator are trained jointly to fool the discriminator. Assuming the discriminator is optimal, the encoder and generator are trained to minimize the Jensen-Shannon divergence [24] between $p(\tilde{\boldsymbol{z}}, \boldsymbol{x})$ and $p(\boldsymbol{z}, \tilde{\boldsymbol{x}})$.

Compared with ALI, our approach also aims to reduce the gap between the joint distribution $p(\boldsymbol{z}, \tilde{\boldsymbol{x}})$ and $p(\tilde{\boldsymbol{z}}, \boldsymbol{x})$. The difference is that we achieve this goal by minimizing the KL-divergence [21] between $p(\boldsymbol{z}, \tilde{\boldsymbol{x}})$ and $p(\tilde{\boldsymbol{z}}, \boldsymbol{x})$ (note that the KL-divergence is not symmetric, so the order does matter). By choosing such a loss function, the discriminator only needs to distinguish between real images and generated images, while in ALI the discriminator is more complicated. We argue that a simpler discriminator structure is advantageous since in ALI the way of combining a pair of a latent vector and an image by concatenation to express the ”joint” relationship may influence the stability of training a GAN. Moreover, a compact structure enables our model to be up-scaled to generate and reconstruct high resolution images. For example, we can progressively increase the resolution using the method introduced in PGGAN [18]. On the contrary, it remains unclear how to apply the progressive growing scheme in ALI where the latent vectors and images are concatenated before being fed into the discriminator.

3. Approach

Figure 2 shows the Reverse-VAE network structure. The generator takes the latent vector $\boldsymbol{z}$ whose elements follow Gaussian distribution $\boldsymbol{z} \sim \mathcal{N}(0, I)$, and generates image $\tilde{\boldsymbol{x}}$. Receiving the generated image $\tilde{\boldsymbol{x}}$, the encoder aims to reconstruct the input latent vector of the generator $\hat{\boldsymbol{z}}$. The discriminator learns to distinguish between the generated image $\tilde{\boldsymbol{x}}$ and the real image $\boldsymbol{x}$. Similar to WGAN [4], the output of the discriminator, $\text{Dis}(\boldsymbol{x})$, is used to calculate the Wasserstein distance, which is also called Earth Mover's Distance [30], .

3.1. Mathematical approach

Let θ denote the parameters for the generator, and ϕ denote the parameters for the encoder. Joint distribution of the latent vector and the image for the generator is expressed by

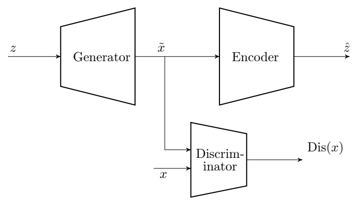


Figure 2: The network structure of the Reverse-VAE model

$p_\theta(\boldsymbol{z}, \boldsymbol{x})$, and joint distribution of the latent vector and the image for the encoder is expressed by $q_\phi(\boldsymbol{z}, \boldsymbol{x})$.

Although KL-Divergence $D_{\text{KL}}(q_\phi(\boldsymbol{z}, \boldsymbol{x}) \| p_\theta(\boldsymbol{z}, \boldsymbol{x}))$ [36] is not mathematically equal to $D_{\text{KL}}(p_\theta(\boldsymbol{z}, \boldsymbol{x}) \| q_\phi(\boldsymbol{z}, \boldsymbol{x}))$, minimizing $D_{\text{KL}}(p_\theta(\boldsymbol{z}, \boldsymbol{x}) \| q_\phi(\boldsymbol{z}, \boldsymbol{x}))$ is leading to the same goal of matching joint distributions $q_\phi(\boldsymbol{z}, \boldsymbol{x})$ and $p_\theta(\boldsymbol{z}, \boldsymbol{x})$.

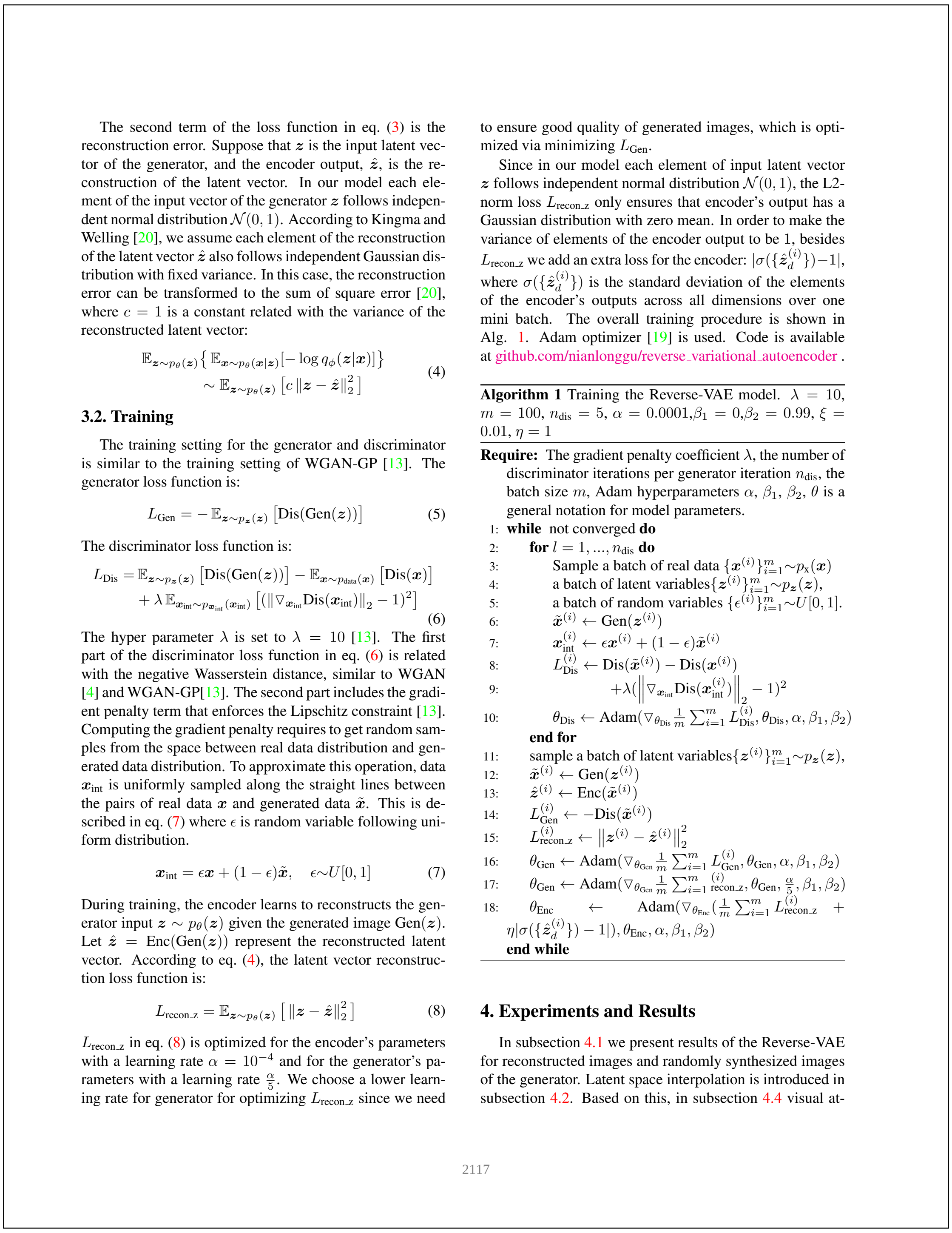
The training goal is chosen to minimize the KL divergence between joint distribution $p_\theta(\boldsymbol{z}, x)$ and $q_\phi(\boldsymbol{z}, x)$:

$$\begin{aligned}
D_{\text{KL}}(p_\theta(\boldsymbol{z}, \boldsymbol{x}) \| q_\phi(\boldsymbol{z}, \boldsymbol{x})) &= \mathbb{E}_{(\boldsymbol{z}, \boldsymbol{x}) \sim p_\theta(\boldsymbol{z}, \boldsymbol{x})} \left[\log \frac{p_\theta(\boldsymbol{z}, \boldsymbol{x})}{q_\phi(\boldsymbol{z}, \boldsymbol{x})} \right] \\
&= \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z})}{q_\phi(\boldsymbol{x}) q_\phi(\boldsymbol{z}|\boldsymbol{x})} \right] \right\} \\
&= \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})}{q_\phi(\boldsymbol{x})} \right] \right. \\
&\quad \left. + \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[-\log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \right] + \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} \left[\log p_\theta(\boldsymbol{z}) \right] \right\} \\
&= \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ D_{\text{KL}}(p_\theta(\boldsymbol{x}|\boldsymbol{z}) \| q_\phi(\boldsymbol{x})) \right. \\
&\quad \left. + \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} [-\log q_\phi(\boldsymbol{z}|\boldsymbol{x})] + \log p_\theta(\boldsymbol{z}) \right\}
\end{aligned} \tag{2}$$

Since the prior distribution of $\boldsymbol{z}$ is fixed during the training process, $\mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \log p_\theta(\boldsymbol{z}) \right\}$ is a constant, it has no contribution to computing the gradient and is neglected here. Therefore, the loss function of the Reverse-VAE model is:

$$\begin{aligned}
L_{\text{Reverse-VAE}} = &\mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ D_{\text{KL}}(p_\theta(\boldsymbol{x}|\boldsymbol{z}) \| q_\phi(\boldsymbol{x})) \right\} \\
&+ \mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} \left\{ \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})} [-\log q_\phi(\boldsymbol{z}|\boldsymbol{x})] \right\}
\end{aligned} \tag{3}$$

The loss function of the Reverse-VAE contains two terms. The first term is the KL divergence between the generator output distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ and the prior distribution $q_\phi(\boldsymbol{x})$ representing the real image data. Similar to the AAE [26], a discriminator is applied to distinguish between generated image (generator output) and the real image. The generator and the discriminator are trained adversarially to minimize the first term $\mathbb{E}_{\boldsymbol{z} \sim p_\theta(\boldsymbol{z})} [D_{\text{KL}}(p_\theta(\boldsymbol{x}|\boldsymbol{z}) \| q_\phi(\boldsymbol{x}))]$ of eq. (3).



The second term of the loss function in eq. (3) is the reconstruction error. Suppose that $\boldsymbol{z}$ is the input latent vector of the generator, and the encoder output, $\hat{\boldsymbol{z}}$, is the reconstruction of the latent vector. In our model each element of the input vector of the generator $\boldsymbol{z}$ follows independent normal distribution $\mathcal{N}(0,1)$. According to Kingma and Welling [20], we assume each element of the reconstruction of the latent vector $\hat{\boldsymbol{z}}$ also follows independent Gaussian distribution with fixed variance. In this case, the reconstruction error can be transformed to the sum of square error [20], where $c = 1$ is a constant related with the variance of the reconstructed latent vector:

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z})}\big\{\,\mathbb{E}_{\boldsymbol{x}\sim p_\theta(\boldsymbol{x}|\boldsymbol{z})}[-\log q_\phi(\boldsymbol{z}|\boldsymbol{x})]\big\} \\ &\qquad \sim \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z})}\left[c\,\|\boldsymbol{z}-\hat{\boldsymbol{z}}\|_2^2\right] \end{aligned} \tag{4}$$

3.2. Training

The training setting for the generator and discriminator is similar to the training setting of WGAN-GP [13]. The generator loss function is:

$$L_{\text{Gen}} = -\,\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\text{Dis}(\text{Gen}(\boldsymbol{z}))\right] \tag{5}$$

The discriminator loss function is:

$$\begin{aligned} L_{\text{Dis}} = &\,\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[\text{Dis}(\text{Gen}(\boldsymbol{z}))\right] - \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[\text{Dis}(\boldsymbol{x})\right] \\ &+\lambda\,\mathbb{E}_{\boldsymbol{x}_{\text{int}}\sim p_{\boldsymbol{x}_{\text{int}}}(\boldsymbol{x}_{\text{int}})}\left[(\|\nabla_{\boldsymbol{x}_{\text{int}}}\text{Dis}(\boldsymbol{x}_{\text{int}})\|_2-1)^2\right] \end{aligned} \tag{6}$$

The hyper parameter λ is set to $\lambda = 10$ [13]. The first part of the discriminator loss function in eq. (6) is related with the negative Wasserstein distance, similar to WGAN [4] and WGAN-GP[13]. The second part includes the gradient penalty term that enforces the Lipschitz constraint [13]. Computing the gradient penalty requires to get random samples from the space between real data distribution and generated data distribution. To approximate this operation, data $\boldsymbol{x}_{\text{int}}$ is uniformly sampled along the straight lines between the pairs of real data $\boldsymbol{x}$ and generated data $\tilde{\boldsymbol{x}}$. This is described in eq. (7) where ϵ is random variable following uniform distribution.

$$\boldsymbol{x}_{\text{int}} = \epsilon\boldsymbol{x} + (1-\epsilon)\tilde{\boldsymbol{x}}, \quad \epsilon\sim U[0,1] \tag{7}$$

During training, the encoder learns to reconstructs the generator input $\boldsymbol{z}\sim p_\theta(\boldsymbol{z})$ given the generated image $\text{Gen}(\boldsymbol{z})$. Let $\hat{\boldsymbol{z}} = \text{Enc}(\text{Gen}(\boldsymbol{z}))$ represent the reconstructed latent vector. According to eq. (4), the latent vector reconstruction loss function is:

$$L_{\text{recon\_z}} = \mathbb{E}_{\boldsymbol{z}\sim p_\theta(\boldsymbol{z})}\left[\,\|\boldsymbol{z}-\hat{\boldsymbol{z}}\|_2^2\,\right] \tag{8}$$

$L_{\text{recon\_z}}$ in eq. (8) is optimized for the encoder's parameters with a learning rate $\alpha = 10^{-4}$ and for the generator's parameters with a learning rate $\frac{\alpha}{5}$. We choose a lower learning rate for generator for optimizing $L_{\text{recon\_z}}$ since we need to ensure good quality of generated images, which is optimized via minimizing L_{Gen}.

Since in our model each element of input latent vector $\boldsymbol{z}$ follows independent normal distribution $\mathcal{N}(0,1)$, the L2-norm loss $L_{\text{recon\_z}}$ only ensures that encoder's output has a Gaussian distribution with zero mean. In order to make the variance of elements of the encoder output to be 1, besides $L_{\text{recon\_z}}$ we add an extra loss for the encoder: $|\sigma(\{\hat{\boldsymbol{z}}_d^{(i)}\})-1|$, where $\sigma(\{\hat{\boldsymbol{z}}_d^{(i)}\})$ is the standard deviation of the elements of the encoder's outputs across all dimensions over one mini batch. The overall training procedure is shown in Alg. 1. Adam optimizer [19] is used. Code is available at github.com/nianlonggu/reverse_variational_autoencoder .

Algorithm 1 Training the Reverse-VAE model. $\lambda = 10$, $m = 100$, $n_{\text{dis}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.99$, $\xi = 0.01$, $\eta = 1$

Require: The gradient penalty coefficient λ, the number of discriminator iterations per generator iteration n_{dis}, the batch size m, Adam hyperparameters α, β_1, β_2, θ is a general notation for model parameters.

```
1:  while not converged do
2:      for $l = 1, ..., n_{\text{dis}}$ do
3:          Sample a batch of real data $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim p_{\text{x}}(\boldsymbol{x})$
4:          a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^m \sim p_{\boldsymbol{z}}(\boldsymbol{z})$,
5:          a batch of random variables $\{\epsilon^{(i)}\}_{i=1}^m \sim U[0,1]$.
6:          $\tilde{\boldsymbol{x}}^{(i)} \leftarrow \text{Gen}(\boldsymbol{z}^{(i)})$
7:          $\boldsymbol{x}_{\text{int}}^{(i)} \leftarrow \epsilon\boldsymbol{x}^{(i)} + (1-\epsilon)\tilde{\boldsymbol{x}}^{(i)}$
8:          $L_{\text{Dis}}^{(i)} \leftarrow \text{Dis}(\tilde{\boldsymbol{x}}^{(i)}) - \text{Dis}(\boldsymbol{x}^{(i)})$
9:                  $+\lambda(\left\|\nabla_{\boldsymbol{x}_{\text{int}}}\text{Dis}(\boldsymbol{x}_{\text{int}}^{(i)})\right\|_2 - 1)^2$
10:         $\theta_{\text{Dis}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Dis}}}\frac{1}{m}\sum_{i=1}^m L_{\text{Dis}}^{(i)}, \theta_{\text{Dis}}, \alpha, \beta_1, \beta_2)$
        end for
11:     sample a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^m \sim p_{\boldsymbol{z}}(\boldsymbol{z})$,
12:     $\tilde{\boldsymbol{x}}^{(i)} \leftarrow \text{Gen}(\boldsymbol{z}^{(i)})$
13:     $\hat{\boldsymbol{z}}^{(i)} \leftarrow \text{Enc}(\tilde{\boldsymbol{x}}^{(i)})$
14:     $L_{\text{Gen}}^{(i)} \leftarrow -\text{Dis}(\tilde{\boldsymbol{x}}^{(i)})$
15:     $L_{\text{recon\_z}}^{(i)} \leftarrow \left\|\boldsymbol{z}^{(i)} - \hat{\boldsymbol{z}}^{(i)}\right\|_2^2$
16:     $\theta_{\text{Gen}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Gen}}}\frac{1}{m}\sum_{i=1}^m L_{\text{Gen}}^{(i)}, \theta_{\text{Gen}}, \alpha, \beta_1, \beta_2)$
17:     $\theta_{\text{Gen}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Gen}}}\frac{1}{m}\sum_{i=1}^m L_{\text{recon\_z}}^{(i)}, \theta_{\text{Gen}}, \frac{\alpha}{5}, \beta_1, \beta_2)$
18:     $\theta_{\text{Enc}} \leftarrow \text{Adam}(\nabla_{\theta_{\text{Enc}}}(\frac{1}{m}\sum_{i=1}^m L_{\text{recon\_z}}^{(i)} + \eta|\sigma(\{\hat{\boldsymbol{z}}_d^{(i)}\}) - 1|), \theta_{\text{Enc}}, \alpha, \beta_1, \beta_2)$
    end while
```

4. Experiments and Results

In subsection 4.1 we present results of the Reverse-VAE for reconstructed images and randomly synthesized images of the generator. Latent space interpolation is introduced in subsection 4.2. Based on this, in subsection 4.4 visual at-

Table 1: FID of Progressive Reverse-VAE on CelebA 256x256 is similar to DCGAN 64x64 and thus, generates good and diverse images. FID of DCGAN trained by a two time-scale update rule (ttur) and of DCGAN from [17].

Method	learning rate	update	FID
DCGAN TTUR [17]	1e-4, 5e-4	225,000	12.5
DCGAN [17]	5e-4	70,000	21.4
PG Reverse-VAE	1e-3	107,496	29.2

tribute manipulation is proposed. Finally, in subsection 4.5 the Reverse-VAE is applied in anomaly detection.

4.1. Random Generation and Image Reconstruction

We trained and tested the Reverse-VAE model on the MNIST [23], the SVHN data set [27] and the CelebA data set [25].

Figures 4a, 4c, 4e show randomly generated images, tested on the MNIST, SVHN and CelebA data set, respectively looking realistic and diverse. In Figures 4b, 4d, 4f the reconstructed images accurately capture the structure, stroke and slope of the digits in MNIST, the center digits as well as the surrounding distracting digits in SVHN, and the main characteristics of faces, including skin color, hair color, hair line, gesture, and facial emotions in CelebA data set, respectively. Based on the results, we conclude that the Reverse-VAE model successfully learns the mapping from the input images to the latent vectors while generating realistic images.

4.2. Latent Space Interpolation

In order to interpolate between two real images, the encoder converts two real images $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ into the corresponding latent vectors $\tilde{\boldsymbol{z}}_1 = \text{Enc}(\boldsymbol{x}_1)$ and $\tilde{\boldsymbol{z}}_2 = \text{Enc}(\boldsymbol{x}_2)$. Then new points $\boldsymbol{z}_{\text{interp}}$ are linearly sampled between the straight line from $\tilde{\boldsymbol{z}}_1$ to $\tilde{\boldsymbol{z}}_2$ with the interpolation factor γ linearly increasing from 0 to 1:

$$\boldsymbol{z}_{\text{interp}} = \gamma \tilde{\boldsymbol{z}}_2 + (1-\gamma)\tilde{\boldsymbol{z}}_1 \tag{9}$$

Afterwards the generator converts the linearly sampled latent vectors to images $\boldsymbol{x}_{\text{interp}} = \text{Gen}(\boldsymbol{z}_{\text{interp}})$ where $\boldsymbol{x}_{\text{interp}}$ are the interpolated images between two real images.

CelebA interpolated images in Figure 5 look realistic implying that the Reverse-VAE learns latent features which generalize well, and that the probability mass does not concentrate around the latent vectors of training samples.

4.3. Progressively Growing (PG) Reverse-VAE

To scale up our model to generate or reconstruct higher resolution images, we adopted the strategy of progressively growing resolution introduced in PGGAN [18]. We train our Reverse-VAE model starting from a very low resolution, 4×4, then we progressively increase the resolution to 8×8 by adding a block of up-sampling and convolution. During the resolution transition stage, a weight factor α increasing from 0 to 1 linearly is used to weight the contribution of the newly added 8×8 block and the previous 4×4 block to the generation of 8×8 images. For the discriminator and the encoder, similar operations of adding a new higher-resolution block and resolution transition are used. We increase the resolution in this manner until reaching the resolution of 256×256, due to a limitation of computation resources.

We also adopted the PGGAN's strategies of stabilizing the training, including minibatch standard deviation, pixelwise normalization, and equalized learning rate. Furthermore, like PGGAN, we remove the sigmoid activation function at the generator's output and rescale the image pixel value into the range of [-1,1]. During the training at each resolution, we are still use the loss functions introduced in Section 3.2 and the training setting is similar to Alg. 1.

Compared with PGGAN, our model has one extra progressively trained encoder, which increases the application scenarios of our model beyond generating HD images. For example, one can reconstruct an input HD image with good accuracy. This enables our model to be used for high resolution image inpainting which means reconstructing lost or deteriorated parts of images. Moreover, our model can easily perform interpolation between two real images using the method in Section 4.2. Figure 3 shows the image random generation, reconstruction, inpainting and morphing results.

The progressively growing Reverse-VAE is shown to be able to generate realistic images and accurately reconstruct features like hair color, skin, facial emotion and gesture in a large image scale. Although in image inpainting the inpainted area has inconsistent brightness, the facial expression looks natural and coincides well with unmasked area. These results further prove the scalability of our model.

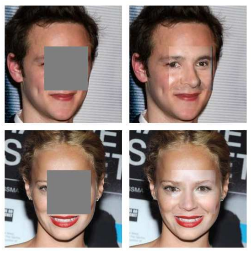
Furthermore, we provide the Fréchet Inception Distance (FID) [17, 7] for random generated images and compare them with other models in Table 1. The smaller FID score is, the higher the quality is and the more diverse the generated images are. The FID of Progressively Growing Reverse-VAE on CelebA 256x256 is similar to DCGAN on CelebA 64x64. The FID confirms that PG Reverse-VAE generates high quality and diverse images.

4.4. Visual Attributes Manipulation

Usage of Feature-wise Reconstruction Loss Although the results of image generation, reconstruction, morphing as well as high resolution image reconstruction show that our model can learn a meaningful embedding and reconstruct the main image features accurately without the feature-wise reconstruction loss $L_{\text{recon\_x}}$ from Equation 1, we do observe

(a) Randomly generated images.

(b) Image reconstructions.


(c) Image inpainting.

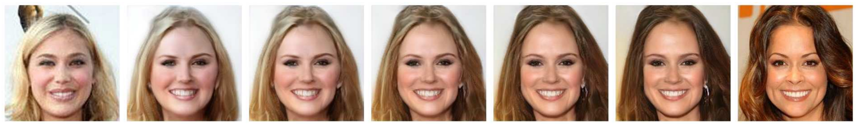

(d) Image morphing.

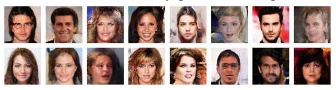
Figure 3: Random image generation, image reconstruction, image inpainting and image morphing using the Progressively Growing Reverse-VAE are tested on CelebA 256×256. For image reconstruction results the first column are input images and the second are reconstructions. For image inpainting we reconstruct the input image in the first column, then only keep the reconstructed area where the mask is, and finally combine it with the input image. For image morhing, the first and last images are real images and the images in between are generated images.

(a) MNIST randomly generated images.

(b) MNIST reconstructions.

(c) SVHN randomly generated images.

(d) SVHN reconstructions.


(e) CelebA randomly generated images.

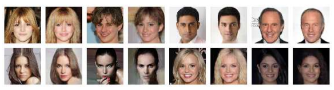

(f) CelebA reconstructions.

Figure 4: In Figures 4a, 4c, 4e randomly generated images and in Figures 4b, 4d, 4f reconstructions on the MNIST, SVHN and CelebA data set, respectively, are shown. For the reconstruction results, odd columns are the original images from test data set and even columns are the corresponding reconstructions.

adding such a loss to the generator improves the reconstruction performance slightly when tested on the CelebA 64×64. This extra loss may force the model to learn to favor a better reconstruction of the detail. In the experiments of visual attributes manipulation (section 4.4) and anomaly detection (section 4.5), we add the loss $L_{\text{recon\_x}}$ by default, since an accurate reconstruction is important for these two tasks.

In contrast to GAN [12], the visual attributes of images can be analyzed in the latent space and particular latent vec-

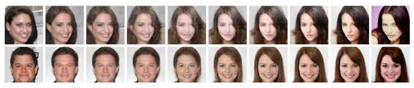


Figure 5: The transition from real image in first column to real image in last column (e.g. woman to man) is based on latent space interpolations with γ increasing from 0 to 1.

tors which represent disentangled visual attributes can be extracted. If we want to give a serious face a smile, it is required to add a visual attribute vector $\boldsymbol{v}_{\text{add-smile}}$ which represents the change from 'serious' to 'smiling' to the latent vector of the serious face.

After training, the CelebA validation data set which includes different celebrity identities is used to compute the visual attribute vectors. Each image is labeled with 40 attributes like hair styles, face emotions and hair colors. For each identity i, the encoder maps each smiling face to a latent vector and then the mean latent vector of smiling faces $\bar{\boldsymbol{z}}^{(i)}_{\text{smiling}}$ is calculated. The same is conducted for the serious face to obtain a mean latent vector of serious faces $\bar{\boldsymbol{z}}^{(i)}_{\text{serious}}$. Then 'serious' latent vector is subtracted from 'smiling' latent vector to obtain latent vector of adding smile $\boldsymbol{v}^{(i)}_{\text{add-smile}}$ for the identity i. Afterwards the latent vector of adding smile is averaged over all possible identities to the visual attribute vector $\bar{v}_{\text{add-smile}}$.

After the encoder processes the corresponding latent vector $\boldsymbol{z}_{\text{serious}}$ for a new image of a serious face $\boldsymbol{x}_{\text{serious}}$, the visual attribute vector $\boldsymbol{v}_{\text{add-smile}}$ is added to the latent vector $\boldsymbol{z}_{\text{serious}}$ to get the transformed latent vector $\boldsymbol{z}_{\text{smiling}}$. Finally, the generator receives $\boldsymbol{z}_{\text{smiling}}$ to generate an image with smiling face $\boldsymbol{x}_{\text{smiling}}$. If the visual attribute vector is disentangled, only the desired visual attribute will be manipulated.

Furthermore, we found that the direction of the visual attribute vector $\bar{v}_{\text{add-smile}}$ determines the type of visual attribute, and the magnitude determines the dominance of the visual attribute. A scale factor α is used to adjust the magnitude of the visual attribute vector. This is achieved by adding the scaled visual attribute vector $\alpha\bar{v}_{\text{add-smile}}$ to $\boldsymbol{z}_{\text{serious}}$ to get the converted latent vector $\boldsymbol{z}_{\text{smiling}}$.

Increasing the scale factor α linearly from 0.5 to 2.5 in Figure 1a, the smile on faces is broadened without influencing other facial attributes. We regard the transition of smile as natural and realistic and suppose that the Reverse-VAE model learns disentangled visual attributes. Manipulated images with 10 visual attributes are shown in Figure 1b.

$$\boldsymbol{z}_{\text{attri,sum}} = \boldsymbol{z}_{\text{original}} + \sum_{j=1}^{m} \alpha_j \bar{v}_{\text{add-attri},j} \quad (10)$$

A set of visual attribute vectors $\{\bar{v}_{\text{add-attri},j}\}_{j=1\ldots m}$ is combined linearly and added to the latent vector of an image in eq. 10, where m is the number of visual attributes and j the attribute index. In figure 6, each α_j is empirically chosen such that the visual attribute is equivalently dominant. Finally, the generator takes the latent vector $\boldsymbol{z}_{\text{attri,sum}}$ to generate the image with desired visual attributes.

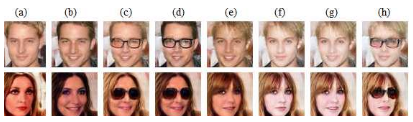


Figure 6: Six different visual attributes are combined: I. Black hair; II. Eyeglasses; III. Smiling; IV. Mouth slightly open; V. Bangs; VI. Pale skin. Each column represents one combination: (a) Original images; (b) Attri. I, III; (c) Attri. II, III; (d) Attri. I, II, III; (e) Attri. IV, V; (f) Attri. V, VI; (g) Attri. IV, V, VI; (h) Attri. II, IV, V, VI.

Different from ALI [8], the Reverse-VAE model is trained without image attributes information. Nevertheless, disentangled visual attributes vectors can be extracted in the latent space learned by the Reverse-VAE, and used for visual attributes manipulation with comparable performance.

Further experiments show that extracting visual attribute vectors without using identity information (such as proposed by Larsen *et al*. [22]) leads to more entangled visual attribute manipulation (e.g gender). In figure 7 adding the visual attribute "blond hair", "heavy makeup" or "pale skin" without identity information to a male face leads to a female face with the desired visual attribute.

4.5. Anomaly Detection

Similar to [3, 37, 31], the image reconstruction error is used to detect anomaly samples. Learning the distribution of training data, the Reverse-VAE can reconstruct the images which are within the distribution of training data with small reconstruction error. For the anomaly images, the reconstruction error is large. Let $\boldsymbol{x}$ denote the input image, $\hat{\boldsymbol{x}}$

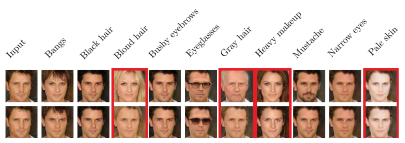


Figure 7: Original images in first column are manipulated without (first row) and with identity information (second row). Visual attribute vectors extracted without using identity information (first row) lead to more entangled visual attribute manipulation.

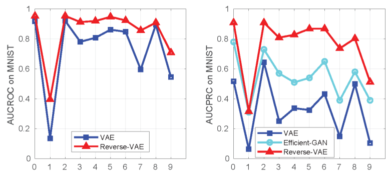


Figure 8: Comparison of anomaly detection performances of VAE [3], Efficient-GAN [37] and Reverse-VAE on the MNIST, evaluated by AUCROC and AUCPRC.

the reconstructed image, and $\mathrm{Dis}_l(x)$ the output of l^{th} layer of the discriminator (here 3rd layer). Reconstruction error $E_{\boldsymbol{x}}$ of input $\boldsymbol{x}$ is defined by:

$$E_{\boldsymbol{x}} = \|\mathrm{Dis}_l(\hat{\boldsymbol{x}}) - \mathrm{Dis}_l(x)\|_2 \tag{11}$$

$E_{\boldsymbol{x}^{(i)}}$ represents the reconstruction error of a sample $\boldsymbol{x}^{(i)}$ from the training set. The anomaly score $A(\boldsymbol{x})$ represents the likelihood that an input image $\boldsymbol{x}$ is an anomaly and is defined by the ratio of number of training samples whose reconstruction error is less than $E_{\boldsymbol{x}}$ to the total number of training samples, whereas $card()$ is the Cardinality sign:

$$A(\boldsymbol{x}) \simeq \frac{card\big(\{\boldsymbol{x}^{(i)} | E_{\boldsymbol{x}^{(i)}} < E_{\boldsymbol{x}}\}\big)}{card\big(\{\boldsymbol{x}^{(i)}\}\big)} \tag{12}$$

The process of anomaly detection is shown in Alg. 2. The Reverse-VAE is evaluated regarding its anomaly detection performance on the MNIST [23].

In **MNIST**, for each type of digits $a \in \{0, 1, ..., 9\}$, we treat digit a as anomaly and all the other digits as normal data. There are 10 different models each trained to detect an anomaly digit respectively. Similar to [3], 80% of the normal data is used for training. The rest 20% of normal data and all the anomaly data are used for testing. Pixels of images are normalized to the range $[0, 1]$. Parameter setting of the generator, discriminator and encoder is the same as for EfficientGAN [37]. The performance of the anomaly detection is evaluated by the area under the curve of the receiver operating characteristic (AUCROC) and the area under the curve of the precision recall curve (AUCPRC).

Figure 8 shows that Reverse-VAE model performs better than VAE [3] and Efficient GAN [37], evaluated by

Algorithm 2 Process of anomaly detection.

```
1: Given input image $\boldsymbol{x}$, compute reconstruction error $E_{\boldsymbol{x}}$.
2: Compute the anomaly score $A(\boldsymbol{x})$ according to eq. 12
3: Select threshold $\epsilon$. $\boldsymbol{x}$ is anomaly when $A(\boldsymbol{x}) > \epsilon$, and
   $\boldsymbol{x}$ is not anomaly when $A(\boldsymbol{x}) \leq \epsilon$.
```

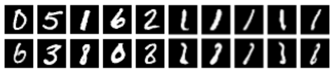


Figure 9: Reconstructions of anomaly digits are given. The first row show anomaly digits and the second row show corresponding reconstructions.

AUCROC. As shown in Figure 9, reconstructions of the anomaly digits resemble samples from normal data set with structural similarity. For example, reconstructions of anomaly digit 7 are mostly 9 or 4. By comparing the reconstructions of anomalies (Figure 9) and normal digits (Figure 4b), we conclude that anomalies can be detected based on reconstruction error, being larger than that of normal digits. Our model has a state-of-the-art performance when evaluated by precision and recall.

Nevertheless, the reconstruction error based strategy is vulnerable to the anomaly images which are structurally simple or similarly appears in other samples. This tendency is also found for VAE and Efficient GAN. Especially detecting anomaly digit 1 is worse than random guess. As shown in Figure 9, the reconstructions of anomaly digit 1 (a) with thick stroke is usually thin version of digit 8 or 3; (b) with normal stroke is usually 7 or 9, since its vertical stroke makes up a large part of digit 7 and 9. The simple structure of digit 1 is present in many other digits, so that it is difficult to detect anomaly digit 1.

5. Conclusion and outlook

We introduced the 'Reverse Variational Autoencoder" (Reverse-VAE) for two applications: visual attribute manipulation and anomaly detection. The Kullback-Leibler divergence between joint latent/data-space distribution of generator and the latent/data-space joint distribution of encoder is minimized during training to learn meaningful mapping from data space to latent space. Based on this mapping both applications are enabled. Desired visual attributes of CelebA images are successfully manipulated. The performance of anomaly detection is competitive with state-of-the-art on MNIST. The anomaly detection can be used as a monitor of a Deep Learning (DL) model trained on the same data as the Reverse-VAE. A positive finding could lead to measures for performing in a safe manner. Furthermore, the good scalability enables our model to be up-scaled for high resolution image visual attribute manipulation which can be used for data augmentation in a practical usage scenario.

References

[1] U. Aftab and G. F. Siddiqui. Big data augmentation with data warehouse: A survey. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2775–2784, Dec 2018. 2

[2] O. A. Aghdam and H. K. Ekenel. Robust deep learning features for face recognition under mismatched conditions. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, May 2018. 1

[3] J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2:1–18, 2015. 7, 8

[4] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *arXiv e-prints*, page arXiv:1701.07875, Jan 2017. 1, 2, 3, 4

[5] S. Burton, L. Gauerhof, B. B. Sethy, I. Habli, and R. Hawkins. Confidence arguments for evidence of performance in machine learning for highly automated driving functions. In *International Conference on Computer Safety, Reliability, and Security*, pages 365–377. Springer, 2019. 1

[6] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *CoRR*, abs/1606.03657, 2016. 2

[7] D. Dowson and B. Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982. 5

[8] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially Learned Inference. *arXiv e-prints*, page arXiv:1606.00704, Jun 2016. 2, 3, 7

[9] A. Fawzi, O. Fawzi, and P. Frossard. Analysis of classifiers' robustness to adversarial perturbations. In *arXiv:1502.02590*, 2015. 1

[10] A. Fawzi, H. Samulowitz, D. Turaga, and P. Frossard. Adaptive data augmentation for image classification. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3688–3692, Sep. 2016. 2

[11] L. Gauerhof, P. Munk, and S. Burton. Structuring validation targets of a machine learning function applied to automated driving. In B. Gallina, A. Skavhaug, and F. Bitsch, editors, *Computer Safety, Reliability, and Security*, pages 45–58, Cham, 2018. Springer International Publishing. 1

[12] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1406.2661, June 2014. 1, 2, 6

[13] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017. 4

[14] M. A. Hanif, F. Khalid, R. V. W. Putra, S. Rehman, and M. Shafique. Robust machine learning systems: Reliability and security for deep neural networks. In *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, pages 257–260, July 2018. 1

[15] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and surface variations. *arXiv preprint arXiv:1807.01697*, 2018. 1

[16] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016. 1

[17] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 5

[18] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 1, 2, 3, 5

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4

[20] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014. 2, 4

[21] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951. 3

[22] A. B. L. Larsen, S. K. Sønderby, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015. 2, 7

[23] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 2, 5, 8

[24] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. 3

[25] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015. 5

[26] A. Makhzani, J. Shlens, N. Jaitly, and I. J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015. 2, 3

[27] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011. 5

[28] X. Peng, Z. Tang, F. Yang, R. S. Feris, and D. Metaxas. Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2226–2234, June 2018. 2

[29] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. Dataset shift in machine learning. MIT Press, 2017. 1

[30] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 59–66. IEEE, 1998. 3

[31] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*, pages 146–157. Springer, 2017. 7

[32] C. Schorn, A. Guntoro, and G. Ascheid. Efficient on-line error detection and mitigation for deep neural network accelerators. In B. Gallina, A. Skavhaug, and F. Bitsch, editors, *Computer Safety, Reliability, and Security*, pages 205–219, Cham, 2018. Springer International Publishing. 1

[33] H. Shi, L. Wang, G. Ding, F. Yang, and X. Li. Data augmentation with improved generative adversarial networks. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 73–78, Aug 2018. 2

[34] J. Shijie, W. Ping, J. Peiyi, and H. Siping. Research on data augmentation for image classification based on convolution neural networks. In *2017 Chinese Automation Congress (CAC)*, pages 4165–4170, Oct 2017. 2

[35] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015. 2

[36] J. Su. Variational inference: A unified framework of generative models and some revelations. *CoRR*, abs/1807.05936, 2018. 3

[37] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018. 7, 8

Testing Deep Learning-based Visual Perception for Automated Driving

STEPHANIE ABRECHT, LYDIA GAUERHOF, CHRISTOPH GLADISCH, KONRAD GROH, CHRISTIAN HEINZEMANN, and MATTHIAS WOEHRLE, Robert Bosch GmbH

Due to the impressive performance of deep neural networks (DNNs) for visual perception, there is an increased demand for their use in automated systems. However, to use deep neural networks in practice, novel approaches are needed, *e.g.*, for testing. In this work, we focus on the question of how to test deep learning-based visual perception functions for automated driving. Classical approaches for testing are not sufficient: A purely statistical approach based on a dataset split is not enough, as testing needs to address various purposes and not only average case performance. Additionally, a complete specification is elusive due to the complexity of the perception task in the open context of automated driving. In this paper, we review and discuss existing work on testing DNNs for visual perception with a special focus on automated driving for test input and test oracle generation as well as test adequacy. We conclude that testing of DNNs in this domain requires several diverse test sets. We show how such tests sets can be constructed based on the presented approaches addressing different purposes based on the presented methods and identify open research questions.

CCS Concepts: • **Computer systems organization** → *Embedded systems*; • **Software and its engineering** → **Software verification and validation**; • **Computing methodologies** → **Computer vision**.

Additional Key Words and Phrases: software testing, deep learning, perception, computer vision, automated driving, autonomous driving

ACM Reference Format:
Stephanie Abrecht, Lydia Gauerhof, Christoph Gladisch, Konrad Groh, Christian Heinzemann, and Matthias Woehrle. 2021. Testing Deep Learning-based Visual Perception for Automated Driving. *ACM Trans. Embedd. Comput. Syst.* 1, 2, Article 3 (April 2021), 27 pages. https://doi.org/10.1145/3450356

1 INTRODUCTION

Deep learning-based approaches have achieved impressive performance results on a wide range of benchmarks in various domains, especially in computer vision (CV, [105]). As industrial application of deep neural networks (DNNs) increases, there is an increased need for their verification and validation (V&V) in safety-critical domains such as (highly) automated driving [16]. In this paper, we focus on practical verification and particularly on functional testing of DNNs used for computer vision tasks in an automotive application.

Consequently, the Software under Test (SuT) is a DNN embedded in an automotive camera with relevant tasks including, *e.g.*, object detection and semantic segmentation. For simplicity, we focus in the following on stateless functions that evaluate each image individually. Janai *et al.* [50] provide a comprehensive survey on the use of computer vision and deep learning functions in automated driving, describe relevant tasks and corresponding methods, datasets and evaluation

Authors' address: Stephanie Abrecht; Lydia Gauerhof; Christoph Gladisch; Konrad Groh; Christian Heinzemann; Matthias Woehrle, firstname.lastname@de.bosch.com, Robert Bosch GmbH, Robert-Bosch-Campus 1, Renningen, 71272.

1539-9087/2021/4-ART3 $15.00
https://doi.org/10.1145/3450356

B.6. Stephanie Abrecht, Lydia Gauerhof, Christoph Gladisch, Konrad Groh, Christian Heinzemann, and Matthias Woehrle. "Testing Deep Learning-based Visual Perception for Automated Driving". In: ACM Transactions on Cyber-Physical Systems (TCPS) *5.4 (2021). ©2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.* DOI: *10.1145/3450356*

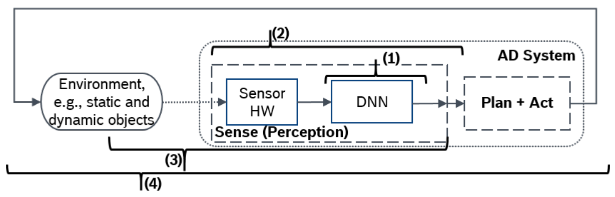


Fig. 1. Overview of our software under test and different test setups.

metrics. While autonomous driving is of major interest for our work, the approaches presented in this work are of relevance for any type of automated vehicle starting from SAE Level 2 (Partial Automation, *i.e.* Advanced Driver Assistance Systems) up to Level 5 fully autonomous systems [79].

In an industrial development process, testing of CV functions is performed throughout various development stages in complementary ways. First, different test methods are used to address various concerns such as checking for implementation errors, data pre-processing concerns, labeling errors, quantization issues, timing and consistency constraints, robustness and satisfaction of requirement specifications. Second, different test setups may be used as shown on four examples marked in Fig. 1 using curly braces. The smallest brace (1) concerns isolated testing of CV functions (DNNs), where images are directly processed by the SuT. Images are subject to effects from the environment as well as the system, *e.g.*, from camera optics or its image sensor [108]. The test setup can also be enlarged, *e.g.*, (2) include the sensor hardware, *e.g.*, in a real hardware-in-the-loop setup. We can use a simulation setup to generate synthetic images based on a given scene description as shown in (3) or perform system testing as shown in (4) [89] requiring closed-loop testing as described in Sec. 3.3.3.

Using DNNs necessitates to consider the *data testing debt* [86]: as we leverage data to train parts of our software system, and thus *"data replaces code"* [86], it seems evident that this necessitates different testing with a particular emphasis on data and its impact. This is challenging as the input space for a typical CV function is vast, especially for high-resolution cameras used in the context of automated driving. In addition, machine learning in general suffers from the "oracle problem", *i.e.* a comprehensive specification of the problem at hand does not exist, as we would not need machine learning if we would know what a model should return for each input datum [109]. The oracle problem is further exacerbated by the open and real-world context of automated driving and the high dimensional as well as unstructured input space of computer vision which leads us to the questions: *How can ML test engineers create relevant and meaningful test data efficiently for deep learning-based visual perception tasks? How can ML test engineers verify relevant properties of the corresponding DNNs? And how can ML test engineers show test adequacy of the available test data?*

In this paper, we review existing works towards practical verification and testing of DNN-based CV functions in an automotive context. Available verification techniques for DNNs are up to now often only applied to the simpler image classification task instead of object detection or semantic segmentation. While image classification task is less relevant for our application domain, the corresponding methods are good starting points for further study. Please note that verification is concerned with building the system right according to specifications, whereas validation is concerned with building the right system. Of course, there is a synergy between the two, *e.g.*, an increase in specification and verification activities may decrease validation efforts (front-loading). We focus on verification, in particular *falsification of the software*, during development, *i.e.* identifying defects in the software early in the development cycle.

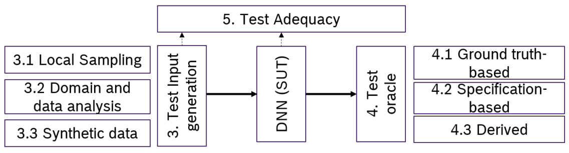


Fig. 2. Structure of the work (building blocks).

As a key difference to related works, we jointly describe approaches from different domains such as software testing, machine learning, computer vision, automated driving, and cyber-physical systems and discuss their applicability to the aforementioned CV tasks in the automotive context. We group the approaches into a test workflow inspired by the work of Zhang *et al.* [109] as shown in Figure 2: Test input generation is discussed in Sec. 3 with local-sampling techniques (3.1), domain and data analysis (3.2) as well as leveraging synthetic data (3.3). This is followed by test oracle generation (Sec. 4) where we discuss ground truth-based (4.1), specification-based (4.2) and derived oracles (4.3). Test adequacy in the form of coverage is described in Sec. 5 using information from the SuT, *e.g.*, for structural coverage (5.1), using the concept of mutation testing (5.2) as well as using a model of the input domain (5.3).

What becomes apparent is that there is not only a single test set for performance, but there is a need for *several test sets each addressing different purposes.* After presenting the building blocks for creating test sets (*cf.* Fig. 2), we conclude with concrete examples of such test sets for DNNs used in automated driving.

2 RELATED WORK

We are interested in the question how one can ensure that a DNN functions as intended in the real world based on finite test sets. We focus on functional properties of the DNN, *i.e.* non-functional characteristics such as timing and hardware constraints are out-of-scope. Previous work has surveyed the general topics of testing of machine learning (ML) software focusing both on implementation and conceptual issues [15]. Zhang *et al.* [109] refer to this as ML testing and provide a comprehensive survey on several applications. Our application domain requires a different view focusing on deep learning and the visual domain. For many testing aspects, there is already a strong discipline for Cyber-Physical Systems (CPS) and (automotive) embedded systems that is too diverse to cover. We refer the interested reader to recent trends in [80]. Here, we focus on novel aspects related to testing of DNNs.

Willers *et al.* [101] provide an overview of safety concerns and possible mitigation approaches for perception tasks. In this work, we identify several testing approaches that can be used for mitigating certain safety concerns such as considering the data distribution as an approximation of the real world (*cf.* Sec. 3.2) and brittleness of DNNs (*cf.* Sec. 3.1). Schwalbe *et al.* [84] survey several methods for safety assurance of ML-based systems in the automated driving context and provide a broad overview of method categories from requirements engineering to validation, while this work focuses particularly on testing and verification.

Borg *et al.* [13] perform a systematic literature review on verification and validation challenges for ML in the automotive industry. However, due to our specific application domain and the focus on verification of perception tasks, our context differs considerably. Salay *et al.* [82] discuss machine learning software in the context of ISO 26262 [47], a standard that concerns functional safety of electrical and/or electronic systems in production automobiles. Salay *et al.* particularly assess "product development at the software level" in part 6 of the standard, which is most relevant for our context, and how it applies to machine learning. They identify that it does not completely address the unique characteristics of ML-based software and the work presented here identifies approaches that could provide a contribution with respect to verification, *e.g.*, concerning coverage (*cf.* Sec.5). Please note that performance limitations such as false predictions of perception functions are not covered by ISO 262626 but rather in the recently published ISO PAS 21448 (Safety of the Intended Functionality) [48]. We refer to [82, 101] for more information on these standards.

Humbatova *et al.* [45] discuss a taxonomy of real faults in deep learning systems based on analyzing open source projects, Q&A sites as well as structured interviews with researchers and practitioners. The taxonomy shows the diverse types of issues that need to be considered in DNN development.

Finally, there exists a large body of work on testing of perception functions in the context of advanced driver assistance systems (ADAS). Feilhauer *et al.* [30] provide a recent overview of approaches used in ADAS testing. They detail on open-loop testing as well as on various forms of closed-loop testing relying on simulation. Open-loop replay tests can leverage large amounts of curated data collected in field trials including previously identified difficult cases. Closed-loop approaches allow testers to evaluate the overall systems reaction considering various aspects. While these techniques remain vital for testing deep learning-based visual perception for automated driving, the required depth and breadth of corresponding test sets increases considerably, *e.g.*, due to additional safety concerns [101].

A first version of this work was published in a workshop to start a discussion by asking several open questions on testing of learned computer vision functions for automated driving [102]. In this paper, we expand upon this work and provide a comprehensive overview on current state of research on obtaining a good test set. In particular, we extend the discussions about test input generation and test oracles and add a novel section on test adequacy presenting several approaches for coverage. We provide a discussion on each individual topic as well as detailed conclusions.

3 TEST INPUT GENERATION

Software testing typically deals with large input spaces through equivalence classes, *e.g.*, separating variables into partitions, where within each partition a common property holds. This is, however, not straightforward for unstructured input data such as images, where (*i*) in pixel space, there are no good ways to specify partitions, except locally around individual samples (*cf.* Sec. 3.1) and (*ii*) when trying to form this partition in a 3D world environment, this would require an accurate mapping from this description to pixel space as input for our algorithms (*cf.* Sec. 3.2).

In the following we describe three approaches for generating test inputs thereby focusing on the question: *How can ML test engineers create relevant and meaningful test data efficiently for deep learning-based visual perception tasks?* First, we consider local sampling around test images (Sec. 3.1) where additional images are created based on existing ones, mostly by small perturbations. Second, we consider analysis approaches that characterize relevant and important factors to be considered for a good test set (Sec. 3.2). Third, we consider testing based on synthetic test data (Sec. 3.3).

3.1 Local sampling around existing images

Local sampling methods leverage existing images and modify them. They are typically used for training to counteract a limited amount of available data. We distinguish two approaches below: First, we describe adversarial data generation where images are modified based on minimal perturbations. Second, we describe a standard approach for dealing with limited data in machine learning, namely data augmentation respectively image augmentation in the CV domain.

3.1.1 Adversarial data generation. For adversarial examples in image classification, an input model is sampled around a given image under the assumption that the labeling stays the same as long as the distance of the new sample is below a selected threshold. A typical norm in the context of (minimal) adversarial examples are L_p norms, *e.g.*, L_∞ used in [94, 103], *i.e.* we consider a neighborhood around each image. Such a metric integrates well into current deep learning frameworks and thus can be used to generate tests efficiently. However, for automated driving other notions of robustness and therefore other distance metrics may be more relevant, *e.g.*, based on noise characteristics of the imager. While most work on adversarial examples is focused on image classification, attacks on object detection and semantic segmentation as well as in the real world have also been studied [4].

Adversarial input models enable verification of satisfiability of a neural network given certain input and output constraints, *i.e.* input constraint for describing the neighborhood of images and output constraints as an invariance property (*cf.* Sec. 4.3). Hence, a symbolic representation of neighborhoods around existing images is used rather than to explicitly sample from it. Liu *et al.* [63] provide a framework for analyzing and comparing various algorithms for verifying neural networks for image classification. Qin *et al.* [75] present a generalization for verification of non-linear properties. They term the current approach of approximating the input space by neighborhoods around test examples, *e.g.*, using an L_p norm, as “weaker verification”, because defining a relevant input set (formally) is extremely difficult. In fact, current approaches to verification of neural networks based on such an input model formulation are more akin to concolic testing in software testing literature [88]. Concolic test generation [88] is a white-box software test generation and analysis technique that allows to incrementally generate a test suite that may ultimately be complete. There is first work to apply concolic testing to deep learning functions: Sun *et al.* [95] use analysis techniques from adversarial robustness for this purpose. A major concern is that for any novel test case generated by a concolic testing tool, we also need to provide a corresponding oracle, which is unclear how to provide due to the “oracle problem” (*cf.* Sec.4).

3.1.2 Image augmentation. Image augmentation for CV functions leverages a large variety of transformations on images like rotations and flips. While its main purpose is multiplication of training data, these transformations can also be leveraged as a basis for testing. Transformation of inputs may be coupled with corresponding transformation of ground truth labels. Some data augmentation techniques may be even amenable to a formal analysis such as the support of rotations in DeepPoly [94].

Augmentation techniques are obviously only an approximation, since transformations in the real world are much more complex than the transformation in the image space, *e.g.*, considering illumination. Additionally, realistic parameterization of image augmentations including thresholds and equivalence classes is a major concern for verification. As an example, we might not use horizontal flipping as this is outside the Operational Design Domain (*cf.* Sec. 3.2) of the function, but may use vertical flipping to approximate left from right hand traffic. There is a difference in the purpose of augmentation for training versus verification. In training, online augmentation with random transformations is used to sample the input space sparsely for a large training set in order to improve generalization. Verification may rather focus on a small set of important test images where augmentation is performed densely, *e.g.*, to better characterize robustness. Here, it is important to know and to control which transformations will be or have been exactly covered and evaluation criteria need to consider whether the resulting image is within the functional specification or used for robustness or negative testing.

3.1.3 Discussion. All methods discussed above try to leverage existing, real images and extend this dataset by small modifications, *i.e.* changing each image, including its corresponding ground truth, in a defined way. This results in sampling the input space locally around existing images.

Some of the discussed methods aspire exhaustive sampling and formal verification, however all results are local around existing images, which results in individual (small) equivalence classes in the vast input space. It is not clear how to check (automatically) that the images in the defined equivalence class are still in distribution. When such tests fail, these are false warnings.

A common idea of these techniques is to leverage input perturbation models that fit to derived test oracles based on invariance, *e.g.*, for adversarials, or equivariance, *e.g.*, for translation augmentation (*cf.* Sec. 4.3). Apart from (formal worst-case) analyses of minimal, additive adversarial perturbations, there are various notions of robustness [32], *e.g.*, datasets considering robustness to corruption and common perturbations [44] as well as computer vision hazards [107] that require different and diverse perturbation models on the input. Robustness (and therefore perturbation models) is stressed by practitioners in the automotive industry as a major challenge in using DNNs and that there are further (empirical) studies needed to define robustness in the application domain [13].

3.2 Domain and data analysis

Domain and data analysis creates and leverages additional information of the relevant context. Figure 3 summarizes the aspects in this area that are discussed in more detail in the remainder of this section. At the center is a definition of an operational design domain (ODD, *cf.* Sec. 3.2.1) in which the CV function needs to operate and under which conditions it needs to work. In the context of automotive and automated driving applications, such factors include environmental conditions (*e.g.*, rain and dusk) as well as the state of the ego-system (*e.g.*, view change due to braking maneuver) and other actors in the environment (*e.g.*, cyclists and pedestrians).

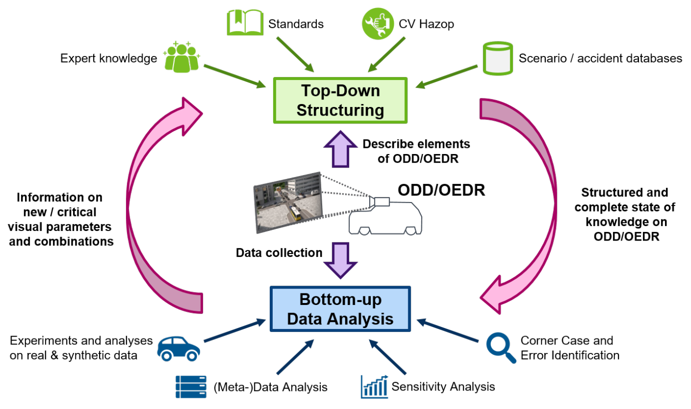

Standards
CV Hazop
Expert knowledge
Scenario / accident databases
Top-Down Structuring
Describe elements of ODD/OEDR
ODD/OEDR
Information on new / critical visual parameters and combinations
Structured and complete state of knowledge on ODD/OEDR
Data collection
Bottom-up Data Analysis
Experiments and analyses on real & synthetic data
Corner Case and Error Identification
(Meta-)Data Analysis
Sensitivity Analysis

Fig. 3. Domain Modeling Overview

Based on an ODD, top-down methods like CV HAZOP (*cf.* Sec. 3.2.2) identify hazards to consider during design and test. Expert knowledge, standards, or scenario databases can be utilized to create semantic domain models (*cf.* Sec. 3.2.3) representing the current state of knowledge on what important semantic features need to be present in a good test set. Most importantly for the following discussion is an identification of nuisance factors [105, 111] and robustness criteria that are not explicitly available in image and label space. This information can be used to define test sets, *e.g.*, by applying a mapping to concrete images [107] or by generating data synthetically (*cf.* Sec. 3.3). Complementary, bottom-up data analyses like sensitivity analyses or corner case identification try to produce additional knowledge on critical visual parameters and combinations of parameters (*cf.* Sec. 3.2.3). This includes an analysis of relevant inputs, *e.g.*, analyze distributions in pixel space, and an analysis of the outputs, *e.g.*, analyze the distribution of ground truth labels. Then, information gathered from bottom-up analyses can be used for updating, extending, enriching and pruning an input domain model, requirements as well as the ODD. Iterating through the loop in Fig. 3 enables to iteratively improve quality and relevance of the test data.

3.2.1 Operational Design Domain (ODD). The Operational Design Domain defines the *where*, (*e.g.*, roadway types) and *when* (*e.g.*, conditions concerning daytime and weather), an automated driving system is designed to operate [2, 57]. Most importantly, a system may exclude conditions from the ODD such as adverse weather conditions such as heavy rain or snowstorms. To date, a complete and detailed ODD definition is elusive due to the long-tail and continuous evolution of the relevant open-world context. However, from a verification perspective, even an incomplete ODD can already be utilized as it documents the current state of knowledge and, particularly, all known corner cases. Therefore, a process as outlined in Fig. 3 is executed iteratively to further concretize and extend the ODD.

The analysis of the ODD is thus very related to the hazard analysis discussed below, as a reduction of the ODD may explicitly result in a reduction of relevant hazards. Similarly, an Object and Event Detection and Response (OEDR) describes the proper handling of external situations that the automated vehicle encounters, including perception [57].

Subsequently, safety requirements on component level are derived from safety goals on system level. Thereby, these requirements shall be linked directly to the ODD [38]. This allows, for example, to test robustness against ODD specific variations. For testing a further concretization such as regarding the test setup (*cf.* Fig. 1) is necessary.

3.2.2 Hazard analysis. CV HAZOP [108] is a hazard analysis approach for computer vision. 1470 hazards are currently available for CV HAZOP [3]. For particular domains as in our case as well as specific computer vision tasks, the analysis needs to be refined [106]. A concrete task in our application domain of interest is presented in context of the WildDash dataset [107]. This work on the construction of a hazard-aware dataset shows that the extensive list of hazards that are relevant for generic computer vision tasks can be broken down to a small number of 9 hazard clusters for a concrete segmentation task. Based on CV HAZOP, Zendel *et al.* discuss different aspects of analyzing image data and in particular negative test cases [106, 107]. For negative test cases, we expect the algorithm to fail, yet with expected behavior, *e.g.*, signaling high uncertainty. Negative test cases additionally provide a means to check that the limits of algorithms as well as the fidelity of the test environment are well-defined. A similar analysis has been performed by Zhang *et al.* [111] for stereo video focusing on specularity, texturelessness, transparency and disparity jumps. The work also describes mapping of hazards to testing via synthetic data (*cf.* Sec. 3.3). Their analysis shows that algorithms that perform better on average are not necessarily better in handling specified hazards. The Data Safety Guidance [85] describes an approach for data management in safety-related systems. It provides guidance relevant to testing such as properties of data that should be considered in the analysis of a data-driven system, such as completeness (*cf.* Sec. 3.2) and fidelity (*cf.* Sec. 3.3) as well as concrete guide words for a data-focused HAZOP analysis similar to CV HAZOP.

A hazard analysis like CV HAZOP and the description of ODD and OEDR share several considerations such as weather, glare and sensor noise [57]. However, there are differences between an ODD/OEDR analysis and a CV HAZOP analysis: CV HAZOP is a generic analysis for *any* CV function, while an ODD/OEDR analysis is focused on a *specific* driving system implementation.

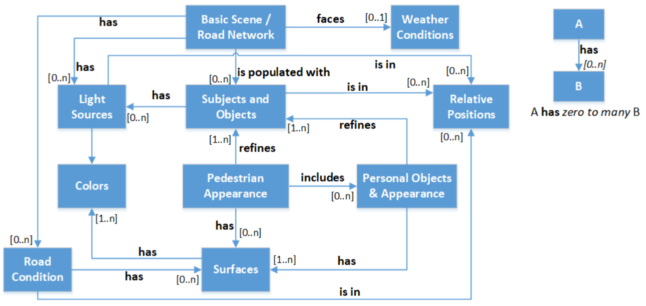


Fig. 4. Conceptual Structure of Semantic Domain Model

On the one side, due to the generic nature of CV HAZOP, many impact factors need to be considered that may not be relevant for a specific application and the system context the application is embedded in. The result is that we may considerably reduce the number of hazards for a given implementation, because hazards may not apply to the system by design or the DNN task [107]. On the other side, a vision function analyzed based on CV HAZOP may still be directly usable even when the scope in ODD or OEDR are extended.

3.2.3 Semantic Domain Models. Semantic domain models are used to further detail the ODD. They capture properties of image contents and metadata of images that go beyond pixel-wise descriptions and sometimes even cannot be directly inferred from the image itself. Such models can be built in a top-down or bottom-up fashion and may form a basis for the structured creation of (test) datasets.

In a **top-down approach**, one tries to structure elements contained in the ODD and to identify properties of these elements and additional effects affecting sensor performance. Inputs for a top-down analysis can be expert knowledge (*e.g.*, from sensor experts), traffic-related databases, public regulations (*e.g.*, highway construction regulations), and (upcoming) standards (*e.g.*, ASAM Open Label [7]). There exist different kinds of databases like scenario databases [29, 113] and accident databases [40]. However, these often do not include perception-specific information, which would be necessary for deriving test information for a computer vision task.

Gladisch *et al.* [42] show how a morphological analysis tool can be used for creating and maintaining a domain model focusing on a road network with different kinds of roads, lane, markings, and traffic regulating elements. In the spirit of equivalence class-based testing, continuous influence parameters are discretized into intervals leading to a discrete combinatorial space. Figure 4 shows a structure of an extended model that additionally includes semantic visual parameters that are important for recognizing objects as their visual appearance including surfaces, colors, weather conditions, and their relative positions. In the case of pedestrians, this includes defining the surface properties and colors of the clothes, which are contained in *Pedestrian Appearance*. What becomes evident is that these combinatorial spaces grow exponentially because, in principle, the interaction of every parameter needs to be checked. While the number of combinations only for the road network is $1.8 \cdot 10^{12}$, it already grows to $7.6 \cdot 10^{34}$ when including a single pedestrian. An approach to deal with this combinatorial explosion with combinatorial testing is presented in Sec. 5.3.

Such top-down methods should be complemented with **bottom-up data analysis**, *e.g.*, in an iterative approach: This may include techniques such as error analysis from machine learning, exploratory data analysis, *e.g.*, for confounding factors, checking the data and label distribution and novelty detection for unique tests. To this end, a combination of targeted data acquisition based on the ODD and random data acquisition for identifying additional domain elements is necessary.

As one example, Cordts *et al.* [22] investigate why performance on Cityscapes changes over the seasons and their analysis concluded that this most likely depends on "softer lighting conditions in the frequently cloudy fall". Many issues such as corrupted images or labels, imbalances [22] and rare data [54], variations, label noise [34] and preprocessing or data quality issues can only be detected by inspecting the data. The same holds true for artifacts within images that are caused by the sensor hardware, *e.g.*, motion blur, or by preprocessing components, *e.g.*, rectification. While approaches such as data inspections, misprediction and error analysis (triage) are vital, they are mostly discussed in practical discussions, *e.g.*, [54, 56], rather than academic venues.

Input domain models can be combined with and expressed in ontologies that enable to capture concepts and relations of the input domain and allow inference of implicit knowledge based on rules [43]. While their use in the context of testing has already been explored for creation of traffic scenarios [8], there are currently no ontology-based approaches for computer vision systems in the automotive domain.

3.2.4 Discussion. All of the methods presented in this section aim at extending and documenting knowledge about the perception function under test. By leveraging this knowledge for the definition of test data, these methods contribute to the definition of relevant test data, whereas we consider relevance of test data as "having a possibility to uncover errors" in this context. While there are no formal guarantees, these methods establish the precondition for finding weak spots resulting from known effects by identifying safety concerns, hazards, nuisance factors, novel aspects, sensor and preprocessing artifacts, and sensitivity to environmental and operational conditions that should be considered in a test set. As a second aspect, documenting and exploiting such knowledge is necessary in a safety-driven development process for arguing safety of a (perception) function [16, 17, 101]. As already indicated in Fig. 3, the analyses and methods discussed in this section need to be applied iteratively such that the knowledge about relevant test data is successively extended.

3.3 Synthetic data

Several works discuss synthetic data and in particular simulation as a key enabler for large-scale testing in the domain of autonomous driving [35, 58]. Borg *et al.* [13] discuss that a promising approach to ML safety engineering is to simulate test cases. There are several benefits of synthetic data including (*i*) flexibility and control of the visual effects and the scene content, (*ii*) massive automatic generation of inputs, (*iii*) inherent availability of precise and unambiguous ground truth, and (*iv*) early availability in the development cycle. In the following, we discuss techniques for generating sets (and sequences) of images with ground truth labeling corresponding to the open-loop testing as shown with the braces (1) – (3) in Figure 1 either via image augmentation and modification approaches (Sec. 3.3.1) or rendering approaches (Sec. 3.3.2). Sec. 3.3.3 discusses closed-loop simulation corresponding to brace (4) in Figure 1.

3.3.1 Image modification approaches. Image modification approaches leverage existing datasets and extend the amount of data and possibly the domain. Compared to Sec. 3.1.2 these are more advanced techniques since these modifications are content-specific. There are several different approaches as to whether (*i*) images are created from learned models whether based on GANs, *e.g.*, DeepRoad [110], or variational autoencoders, *e.g.*, [18], (*ii*) augmented with sensor effects such as chromatic aberration and blur [19] and other style transfer approaches, (*iii*) perturbed for robustness testing [73, 93], and (*iv*) augmented with additional relevant agents and objects, *e.g.*, for urban driving scenes [5]).

B.6. Stephanie Abrecht, Lydia Gauerhof, Christoph Gladisch, Konrad Groh, Christian Heinzemann, and Matthias Woehrle. "Testing Deep Learning-based Visual Perception for Automated Driving". In: ACM Transactions on Cyber-Physical Systems (TCPS) *5.4 (2021). ©2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.* DOI: 10.1145/3450356

The focus of these methods is typically on training and showing a benefit of leveraging synthetic data. As an example, Nowruzi *et al.* [70] evaluate how to combine synthetic and real datasets for car and person detection. Waymo engineers [21] describe improvements for AD computer vision tasks by using Randaugment [23]. Considerations for verification may be different: instead of improving average-case behavior over a realistic distribution, testers may rather be interested in improving the least worst-case behavior [32]. Additionally, depending on the type of property, tests may be interested in addressing the content gap and/or the appearance gap of a simulation [53].

3.3.2 3D Rendering approaches. A wide range of computer graphics technologies and ecosystems exist. This makes a proper use and a corresponding evaluation for a V&V task difficult for non-experts in computer graphics. The two main rendering approaches are scanline rendering and raytracing. Traditionally the former is better for performance and the latter for quality, however, the combination of many other technologies has a bigger impact on quality and performance. These are technologies for (*i*) object and scene geometry, (*ii*) textures and materials, (*iii*) lighting, in particular global illumination, camera and color formats (*iv*) animations and physics, and (*i*) simulation of various visual phenomena. Each of these subareas is a research topic on its own and a combination of these technologies allows achieving photo-realistic results. A concrete example for this is the Synscapes dataset leveraging *advanced* 3D rendering techniques where the authors show the application of synthetic data with a high level of realism for both training and evaluation [104]. Since setting up such an advanced 3D rending pipeline is difficult, many research works rely on open-source simulators such as CARLA [25] and AirSim [91]. These simulators provide (*i*) an easy-to-use programming interface to generate images and control complex 3D graphics engines, (*ii*) free content in the form of 3D environments and objects, (*iii*) a setup suitable for generating images relevant for vision-based perception for automated vehicles (*iv*) generate ground truth which is non-trivial to get when starting with a 3D rendering tool.

Computer graphics are used in several works for training and testing of DNN-based computer vision functions, such as (*i*) a 3D World simulation for pedestrian detection [74], (*ii*) training and testing object detectors with virtual images [96], (*iii*) analyzing hazards for stereo vision algorithms [111], and (*iv*) using probabilistic scene grammars combined with learning an adaption of generated scenes for dataset and task-specific synthetic content generation [53]. A higher-level approach for training and testing of perception systems based on these frameworks and game engines is a language such as Scenic [33]. It focuses on variations on a domain-level (*cf.* Sec 3.2.3) by providing a domain-specific language on the level of scenarios and a probabilistic programming approach. An overview of specific simulators for automated vehicles can be found in Table 6 in [78].

Research and methods in computer graphics are not only useful for synthetic data generation for testing computer vision. The computer graphics field has a good understanding of the ingredients that constitute an image and this knowledge could be a basis for semantic domain models and for details on the ODD as described above in Sec. 3.2. For instance, the distinction between textures and geometry yields general concepts, which are useful for describing properties of vision. As an example, Zoox [114] mentions that it is not necessary to always classify objects correctly but it is important not to miss any relevant object such. This can be interpreted as not to miss any critical geometry, *e.g.*, anything "person-like". In computer graphics more concepts and taxonomies exist which may be beneficial to derive a taxonomy for the ODD and testing parameters.

3.3.3 Closed-loop dynamic environment simulation. So far we have focused on open-loop simulation for generating a set of images with ground truth or sequences thereof. For simulation in the AD domain often the focus is *Plan* + *Act* (cf. Figure 1), *e.g.*, for search-based testing [41]. However, correct functioning of *Plan* + *Act* alone does not ensure safety of the system if perfect perception is assumed as their input. Hence, integration testing is necessary and requires a test setup of *Plan*+*Act* together with perception (brace (4) in Figure 1) in order to account for uncertainty and statistical results from perception. For developing and testing *Sense* + *Plan* + *Act* (brace (4) Figure 1) a closed-loop test setup is required. While testing may be performed in the target system, a simulation-based environment is important, *e.g.*, for rapid progress in development [14]. CARLA [25] and AirSim [91] are two simulators used for closed-loop simulation including perception. As an example, in the CARLA simulation challenge autonomous driving agents are tested in a virtual environment [77]. Closed-loop tests with a perception focus are typically restricted to simulations of (short) scenarios. An industrial example is the standardization activity on OpenSCENARIO to "describe complex, synchronized maneuvers that involve multiple entities like vehicles, pedestrians and other traffic participants" [31]. Thus, agent-based simulations that target the (long-term) interaction of traffic participants are typically not the focus and very challenging, see *e.g.*, [10]. Nevertheless, variability in scenarios needs to be considered. For example, when using Scenic [33], a description of a scenario (class) actually results in several simulations sampled from the distribution of a scenario including variations of the interactions among agents.

Given such a closed-loop simulation, falsification techniques from the CPS domain such as search-based testing [41] can be used also for systems that include machine-learning based perception. Dreossi *et al.* [26] describe such a compositional approach. They propose a decomposition in order to be able to analyze the system behavior and the perception performance separately and improve the falsification process. Cruise describes how they use simulations for developer and integration testing [14]. While they use different simulation models depending on the software under test, some simulations of their tool Matrix provide a full set of perception sensor inputs (camera, radar and lidar). As discussed in [14], there is always a trade-off between fidelity and run-time performance. As such (*i*) the level of accuracy will differ and (*ii*) an analysis of the suitability of the resulting sensor data needs to be considered as discussed in the following.

3.3.4 Discussion. Synthetic data is an important building block for training and testing DNNs. Leveraging synthetic data, *e.g.*, via simulation, is a commonly-used technique in testing automotive embedded systems. A discussion in the context of simulation models for hardware-in-the-loop systems can be found in [46]. Hutter's focus is on classical automotive systems such as vehicle stability control, but the discussion of model affordances (*What sort of functionality should the model provide?*) and fidelity (*What fidelity does the test required to decide whether a specification is satisfied in the actual system?*) is valuable for any simulation-based approach and needs to be decided for each verification task [14].

A specific concern when using synthetic data is the required fidelity such that results transfer to the real world [58]. The problem is not specific to synthetic images or computer vision but applies generally to testing based on models and abstractions, *e.g.*, when not using the target system in its ODD. The problem of false alarms, *i.e.* a test on synthetic data fails but there is no issue in the real world, and missed violations, *i.e.* the test on synthetic data passes but there exists an issue in the real world, concerns many testing tools and has recently been discussed in the context of static analysis results by Meyer [68]. Although from a verification perspective false alarms are not harmful, "false alarms kill an analyzer" [68] *w.r.t.* user acceptance. Missed violations may be catastrophic and thus need to be specifically considered in a safety-related process step. The general approach to address this problem is validation and conformance testing of the synthetic data generator using a real system in the target environment and corresponding data. Comparison of results based on synthetic and real images is typically performed in previous works and several authors have reported similar *average* performance between their simulations and related datasets [5, 67, 74, 96, 111]. However, depending on the use case, average performance may not be sufficient.

A particular concern in validation is that the simulation model accurately reflects the considered performance of modeled components for the system property to be checked [46, 58]. Real data allows engineers to check assumptions of simulation models, *e.g.*, on model simplifications. Dedicated back-to-back comparisons on controlled (closed course) tests and corresponding simulation runs enables detailed study of simulation inaccuracies and their effect on the properties to be checked. Waymo describes using data from closed course testing for improving and validating simulation models [100]. Simulation validation has mostly been discussed in the context of driving dynamics, *e.g.*, [64], hence are based on different measures of simulation fidelity than required for perception models.

In summary, synthetic data generators with a sufficient fidelity may have significant cost in creation, maintenance, validation and execution. However, a benefit of such synthetic data generators is that testing can be scaled economically and allows to test systematically [105], *e.g.*, based on a semantic domain model (*cf.* Sec. 3.2.3).

4 TEST ORACLE GENERATION

Testing machine learning suffers from the oracle problem [11, 109]. For many relevant inputs, it is impossible to specify the correct prediction [24] in a general form. It is challenging to write an oracle for all possible samples, as features may be ambiguous and entangled in the input space. This is why (supervised) machine learning relies on labeled data. Labels for an image are point-wise specifications, *i.e.* , a description of the expected results for exactly one data sample. While it is possible to express (human-readable) requirements for expected outputs ("A person should be detected"), a test oracle requires to write these requirements concretely in a machine-readable and machine checkable form. Concretely, we typically cannot describe images in a given ODD in a machine-readable format as described above in Section 3 and similarly, we also cannot completely specify the expected output in a general, machine-checkable form (what about partially occluded persons or persons far away?) When labeling individual images of a dataset, the lack of a general oracle can be partly resolved. However the issues of dataset-based approach such as representativeness, coverage, unintended correlations and a loose relation to the ODD remain.

So, *how can ML test engineers verify relevant properties of the corresponding DNNs?* Obviously, for single images this can be manually or semi-automatically performed with labeling, but this necessitates a scalable approach to generate high quality labeling data. We detail on labeling in Sec. 4.1. As Barr *et al.* [11] describe, there are several ways to deal with missing oracles. We discuss evaluation functions such as properties, metrics and derived specifications in Sec. 4.2. Here, we need to differentiate the general approach of testing guided by a particular test purpose such as checking for (*i*) average case behavior, *e.g.*, to determine mean Intersection-over-Union on a test set [22], (*ii*) exceptional behavior, *e.g.*, checking uncertainty assessment for image data outside the training distribution, and (*iii*) specifications in particular use cases such as detection of relevant, yet far-away objects. Note that recent academic work has focused on comparing average case behavior based on cost metrics - mainly to evaluate competing designs, *i.e.* case (*i*) above. Verification and testing are typically concerned with (worst-case) behavior *w.r.t.* specific properties. As an example from industry, Karpathy [54] discuss how individual unit tests are grown with a curated set of dedicated predicates for individual images.

4.1 Oracles from ground truth and meta information

Deep learning in computer vision mostly relies on supervised learning, *i.e.* ground truth labels are provided for training. Predictions are compared to ground truth labels based on task-dependent cost metrics such as intersection-over-union (IoU). One of the main advantages and reasons for using synthetic data is automatic generation of ground truth along with the generated input (*cf.* Sec. 3.3).

The quality of ground truth may be superior to manual labeling because of its underlying algorithmic computation. Therefore ground truth is consistent and precise, i.e. pixel-precise, *e.g.*, when creating semantic segmentation maps. In contrast, for acquired real data, creation of ground truth is typically a manual task and often supported by additional measurements, *e.g.*, lidar that help with object detection and tracking [60].

Getting data with matching ground truth is necessary for training, validation and verification and often one of the most expensive and time-consuming tasks [32, 53] and a key bottleneck [76]. As an example, fine segmentation maps for Cityscapes required more than 1.5 hours per single image on average [22]. We refer to literature from computer vision [50, 67, 105] and benchmarks for automated driving [22, 50] for data collection and labeling. There are similarities between manual labeling and manual software testing as both rely on a specification subject to interpretation by humans [32]. One difference is that for V&V we typically want to use high-quality ground truth, as the quality of labels is a major safety concern for DNNs [101], while training may also rely on weak signals based on automatic labeling [76].

As in any test setup, if a test execution fails, the error may be due to the implementation or due to the test. For reference-based tests, the actual reference, *i.e.* in our case ground truth labels, may be at fault. There are several sources of errors and label noise [34] including errors of automatic pre-labeling, errors in the labeling specification given to labelers and human errors in labeling [22, 32]. Concrete challenges in labeling are domain-specific. As an example, in automated driving a fine-grained classification might not be necessary (*cf.* 30 classes in Cityscapes [22]), yet there may be some subtle interpretations, *e.g.*, of drivable space on road boundaries. To ensure the high-quality labeling required for V&V, quality control of labeling results are necessary and important.

Gauerhof *et al.* [37] focus on the elicitation of deep learning safety requirements including data requirements. According to the desired properties, also called desiderata for data management *relevant* and *complete*, data requirements refer to data defined in ODD and variants occurring due to other components in the system, *e.g.*, sensor properties. To satisfy the desideratum *accurate*, data requirements are specified regarding the quality of labeling. To satisfy the desideratum *balanced*, the data shall have a comparable representation of samples for each relevant class and feature, meaning that any class must not be under-represented with respect to the other classes or features. Especially testing may reveal under-represented data in training, if, for example, robustness against a special feature (or meta information) is not adequately existent. One consequence could be to enrich the data, especially training data, so that after retraining the robustness requirement is satisfied. While data requirements might only partly capture the dependence of a DNN behavior on the data, testing shall exhibit intended and unintended correlations of both.

Discussion. Labeled data is required for both training and testing. While training may also benefit from data with noisy labels or even use unsupervised and semi-supervised techniques, testing and verification depend on the availability of high quality labels from the target system in order to ensure a realistic performance evaluation. If label quality is not sufficient, the model might suffer from undetected insufficiencies that will only show later in the field.

Even though synthetic data already comes with automatically generated ground truth and can be scaled economically (*cf.* Sec. 3.3), it should only be used for testing after having shown its validity with respect to real data. Hence, real data from the target domain will always be needed at least for conformance testing and validation.

For detailed evaluation further task-specific labeling and metadata may be necessary, *e.g.*, for categorization into a semantic domain model as shown in Fig. 4, and especially considering domain-specific evaluation metrics that require fine-grained details about the environment, *e.g.*, for determining relevance of objects as discussed in Sec. 4.2.2.

4.2 Specification-based oracles

Providing a full set of specifications for a perception function is unrealistic given the open context of the problem. However, we can formulate (partial) specifications for various properties and sub-tasks [81]. This is beneficial as such specifications (*i*) strengthen our safety argumentation and (*ii*) make verification more efficient. Although a complete decomposition into subtasks is not feasible for a perception function, each specified property and sub-task helps to find causes for failures and unintended behavior. Bottom-up approaches (Sec. 4.2.1) start from the perception function design and implementation and specify lower-level properties such as invariance under perturbation. Top-down approaches (Sec. 4.2.2) specify properties derived from the context, ODD, system design and decomposition, *e.g.*, fine-granular performance specification in partitions of the input space. Due to the nature of the environmental perception task, deriving specifications is not a one-time approach, but requires iterations and interaction between these bottom-up and top-down approaches (*cf.* Fig. 3).

4.2.1 Bottom-up approach. Ground truth is a reference, a point-wise specification, with predictions assumed to be (approximately) equivalent. Ground truth in combination with image augmentation frameworks as described above (*cf.* Sec. 3.1.2), can also be used to describe local invariance and equivariance properties:

(*i*) *Local invariance*: Small changes on the input do not cause a change on the output. An example is the setup for minimal adversarial examples in image classification as described above, where around images the (top-1) classification shall remain the same. We can also relax this property and for example only check that after input transformation, the labeled class is contained within the top-5 predictions.

(*ii*) *Equivariance*: A change on the input causes an equivalent change on the output. An example in the context of image augmentation for semantic segmentation is the translation and rotation of a segmentation mask alongside its input image. Leveraging equivariance and invariance as test oracles can help us to reveal which patterns a DNN might learn based on the given data and their corresponding artifacts. Thereby, spurious correlations may be revealed.

(*iii*) *Spatio-temporal*: When considering sequences of images, spatio-temporal properties of detected objects can be checked for consistency. In order to achieve this, object behavior must be translated to image space, *e.g.*, to determine bounding box translations across frames. Balakrishnan *et al.* [9] describe a formal language called Timed Quality Temporal Logic (TQTL) to describe such spatio-temporal properties. While they only approximate translations in image space, this can already identify relevant misclassification for popular convolutional neural network architectures used for real-time object detection. Such properties are closely related to using consistency properties in machine learning. A recent approach by Varghese *et al.* [99] describes temporal consistency across frames and improves the approximation of changes in image space by leveraging optical flow. On this example, we can see the relation between consistency metrics used in (unsupervised) learning and property specifications used in verification. Properties determined from bottom-up analysis should be included into requirements and thus, enrich the top-down determined specification with machine learning specific, safety relevant properties.

4.2.2 Top-down approach. After structuring the ODD (*cf.* Sec. 3.2.1), analyzing hazards (*cf.* Sec. 3.2.2) and describing the ODD in a more detailed way, such as in a semantic domain model (*cf.* Sec. 3.2.3), the requirements elicitation is conducted. Thereby, it is challenging to provide a traceable link between system safety requirements and deep learning safety requirements, while providing verifiable deep learning safety requirements. Gauerhof *et al.* [37] focus on this issue. To support a safety case, they define a set of desired properties (desiderata) for data management

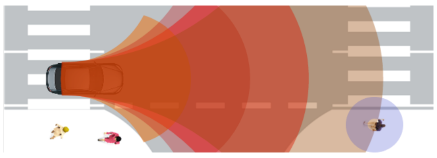


Fig. 5. Relevance of a moving pedestrian (movement prediction in blue) for the vehicle moving from left to right (movement predictions in red).

(*cf.* Sec. 4.1) and model learning (performant, robust). To ensure the desiderata are satisfied, safety requirements are defined for data management and model learning. By doing so, a traceable link between system safety requirements and deep learning safety requirements is established.

Frtunikj *et al.* [35] discuss a refinement of performance metrics to application-specific quantities depending on safety requirements: This may include the consideration of the environment such as weather and the context of the ego vehicle, *e.g.*, in the form of distance-based mAP (mean average precision). Metrics with more structure are intuitively appealing, since speed and distance are relevant parameters in following stages of the AD functional chain, *e.g.*, for computing threat metrics of a planner. The context-dependent performance metrics discussed in [35] are obviously linked to a domain analysis (*cf.* Sec. 3.2). Additionally, within requirements each metric needs a corresponding acceptance threshold or area. Detailed metrics also necessitate availability of the corresponding meta-information relating to the ODD or requirement elements, *e.g.*, object distances for the metric discussed above.

Seshia *et al.* [89] survey the landscape of formal specifications for DNNs. The paper mentions that a complete end-to-end perspective may be appealing, since system level constraints, requirements and rules are more intuitive and easier to specify on a system level. Moreover, breaking down constraints and requirements across the functional chain is a challenging task [98]. However, (*i*) for practical formal verification a decomposition is necessary and (*ii*) end-to-end test setups such as (4) in Figure 1 are expensive and require a check for validity, *e.g.*, for a simulation environment as described in Sec. 3.3.3.

As part of the requirements for a driving task, the **relevance** of individual objects may be considered to assure safety-relevant properties. As an example, Bolte *et al.* define corner cases in terms of relevance as follows [12]: "A corner case is given, if there is a non-predictable (*i*) relevant object class in (*ii*) a relevant location." Non-predictable refers to a "misprediction" of the DNN. Relevance in the context of the driving task could be either a static element such as a traffic sign relevant for the determination of the scene and relevant traffic rules or a (dynamic) traffic participant such as a pedestrian. In the latter case, a relevant pedestrian could potentially have some impact on the movement of the AV depending on both locations.

The intuition is that a faraway pedestrian that cannot be reached within a specific time frame that depends on the ego vehicle's speed has less immediate impact on planning and most likely no impact on immediate safety of the system [101]. Some works consider this as individual requirements in particular situations [37], while other works integrate a relevance metric for that purpose [12]. Often, relevance is used synonymously with distance [35] derived via measurement using a reference sensor, *e.g.*, using a lidar as in [39], or approximated by proxy metrics on the image space, *e.g.*, an object's distance from the bottom of the image. A further refinement of relevance is to consider the actual movement of traffic participants for relevance propagations as shown in Fig. 5. As shown for the ego vehicle, relevance is not purely derived on distance, but may additionally consider the actual movement possibilities of traffic participants over time. This results in a refined evaluation metric that specifically defines relevant locations for each relevant object class type individually. This in turn allows a focus on corner cases as defined above, which may help to mitigate the safety concern of "Unknown behavior in rare critical situations" [101].

Instead of a complete specification that is challenging, a partial specification might strengthen safety assurance [81] and thus, testing. One approach is contract-based design (CBD), *e.g.*, [83], that enables refinement checks, enhances the validity of requirements and can be included in test derivation. CBD requires to define what each component guarantees, provided its environment satisfies the given assumptions [17, 83]. Contracts are gained by formalizing informal requirements of components. Contracts are used for refinement checks, *e.g.*, to check properties of the overall system based on components[52]. Additionally, contracts can be used for (automatic) test generation [61]. In contrast to CBD for cyber-physical systems [83], contracts for perception and machine learning need to deal with ML idiosyncrasies such as inherent uncertainty. This may necessitate new approaches in formalizing and analyzing contracts, such as the Chance Constraint Temporal Logic to include chance constraints as predicates [51] or TQTL for spatio-temporal properties [9] mentioned above. Guarantees may be formulated based on the DNN performance and conditional on partitions of the input space [17], *e.g.*, by considering a semantic domain model as described in Sec. 3.2.3. Contracts may also include derived properties such as equivariance (*cf.* Sec. 4.3).

4.2.3 Discussion. Specifications of perception functions are tightly coupled to an input. On the one hand, we condition on partitions of the input space to characterize the functions behavior in detail (*cf.* Sec. 3.2.3). On the other hand, within each partition we need to provide diversity in order to allow for robust evaluation within the partition. These input partitions are typically based on human intuitions, while a DNN learns its partitioning from the input data and thus may be different. Of importance is to find systematic and reproducible errors by testing and to find reliable, effective metrics for testing. Requirements shall utilize these metrics such that test goals and setups are unambiguous Moreover, metrics that are automatically analyzable and approaches that do not require ground truth reduce the effort for testing. Requirements including relevance of objects refine metrics so that test results are more meaningful for the application domain, *e.g.*, to relate which objects in the environment of a vehicle are of high importance for a planned driving task and which are not important. To take system-specific as well deep learning-specific properties into account for requirements, top-down as well as bottom-up approaches are needed.

4.3 Derived oracles

There has been work on differential and metamorphic testing of machine learning functions, both so-called derived specifications [11]. First, differential testing or the more general variant of n-version testing is a derived test oracle [11] where the results of n versions of the same function are compared with each other. Here we rely on a comparison of functions that observably differ in their implementations. This can be either based on different implementation approaches or on evolutions of a single implementation, *e.g.*, for regression testing. DeepXplore [72] uses differential testing for different applications including a simple example from the automated driving domain. The use of redundant paths in safety-critical applications, whether through different sensing modalities or redundant implementations, supports a use of differential testing and can help to identify inconsistencies, but not common weak spots. As an example, Karpathy [54] describes differential testing of traffic sign recognition based on images compared to traffic signs annotated in a map.

There are several limitations to consider:

- Multiple implementations are needed that feature some sort of diversity in order to have variance in predictions.

- In order to receive meaningful results, all implementations should have strong performance to avoid finding easy cases only.
- The source of a difference in observable behavior is unknown and could be that (*i*) one implementation is wrong, (*ii*) both implementations are wrong (although unlikely), (*iii*) the input for differential testing was outside the specified input domain and (*iv*) there is inherent uncertainty in the prediction.

Second, metamorphic testing [87] uses metamorphic relations (MRs) that check the relative change between different executions of the same SW under test. A simple example of a MR is a difficult-to-specify function with periodicity (in the literature a simple sine function is used for illustration purposes). While an absolute specification of an output may be difficult to formulate for a concrete input x, we know that if we have given the function evaluation $f(x)$, we can directly determine the outputs at periodic inputs $f(x + n * period) = f(x), \forall n \in \mathbb{N}$. MRs may describe a local (epsilon) invariance property and as such currently formulated adversarial robustness properties as well as augmentation invariance are using metamorphic relations. Local invariance is also mainly used in previous work on metamorphic testing, *e.g.*, in DeepTest [97] and DeepRoad [110] where it is checked whether a projected driving vector remains invariant within some ϵ bound.

Metamorphic relations can however be used for more general specifications such as equivariance properties. As an example for equivariance, a translational change on the input should result in an equivalent translation of bounding boxes of an object detector. As such, the combination of concrete tests, *i.e.* data points labeled with ground truth, and relations that capture relevant oracles around the concrete tests seems suitable for testing machine learning classifiers. Some metamorphic properties are described by Dwarakanath *et al.* [28] in the context of testing for implementation bugs. As such, the focus is not specified behavior, but common properties that may help to detect generic implementation bugs, in this case exemplified via mutation testing of the underlying source code.[1] Note that any derived specification that only approximates a real requirement is susceptible to false alarms and missed violations as described for synthetic data (*cf.* Sec. 3.3).

Discussion. Any machine learning application inherently suffers from the "oracle problem". This is exacerbated in deep learning where additionally features are learned creating large black boxes that are hard to understand and therefore test. Besides top-down determined specification, we can derive oracles, *e.g.*, for robustness, invariance or equivariance properties. As brittleness of DNNs is a prime safety concern [101], oracles for robustness is an important research topic.

Given the discussion, it is evident that derived specifications have a relation to making DNNs understandable as both have the goal of opening up the black box. As an example, the context removal technique by Shetty *et al.* [93] to quantify and control the effects of context is a method for determining robustness that can be leveraged for verification. As we can see on the example, derived specifications are often tied to input generation techniques, *e.g.*, as discussed in Sec. 3.1 and Sec. 3.3.1. As an example, Dreossi *et al.* [27] present an overview of different robustness formulation including an interesting comparison between different adversary input generation techniques under the same general notion of robustness by considering the formulation of the input perturbation (*cf.* Sec. 3.1) and the corresponding derived oracle on the output.

5 TEST ADEQUACY EVALUATION

In this section, we answer the final question *how can ML test engineers show test adequacy of the available test data?* To this end, we discuss test adequacy and in particular different forms of coverage, since coverage is widely-used in industry as a quantitative measurement of test

[1]Machine learning specific mutants were created, *e.g.*, changing the loss function.

adequacy [109]. Well-known are structural coverage metrics based on code such as branch coverage. Such structural metrics are used when testing small software units. ISO 26262 [47] discusses the above mentioned coverage metrics for software unit verification of safety-critical code. However, coverage criteria are not restricted to code and are selected based on the corresponding test level. ISO 26262 discusses function and call coverage as structural coverage criteria at the software architectural level and similarly coverage of requirements and respective equivalence classes on an embedded software level. Hence, on a lower level, white-box, structural coverage metrics are used that consider the implementation and design, while on a high functional level black-box coverage criteria are used that provide information on what the software is supposed to do. In summary, a coverage metric supports the testing process by providing a quantitative measure that must be adequate for the test level and purpose.

In the following, we discuss coverage metrics bottom-up and start with structural coverage metrics in Sec. 5.1. Then we look at mutation coverage in Sec. 5.2. We finally present coverage from the input domain perspective (Sec. 5.3).

5.1 Structural coverage for neural networks

There have been several works on coverage for neural networks. Most coverage metrics are fundamentally based on structural coverage and formulate metrics based on neuron activations [62, 65, 72, 95]. As described in previous work [1], it is imperative to have a clear definition of (*i*) the definition of an individual coverage element ("neuron"), (*ii*) how the corresponding activation is split into equivalence classes and (*iii*) the aggregation into an overall metric. There is a large number of different approaches that can be and have been defined. An overview is provided in [1] and [95]. Abrecht *et al.* [1] compare differences in neuron and activation definition and report the resulting differences in coverage elements that can be several orders of magnitude depending on DNN architecture. Sun *et al.* [95] discuss the relation of structural coverage criteria to code coverage and formally study the subsumption relationship between several structural coverage criteria for DNNs. There is some criticism towards such structural coverage metrics. As [1] points out, test generation based on structural coverage may not perform better than standard augmentation techniques for test generation. Li *et al.* [62] discuss that many structural coverage metrics have been only used to evaluate effectiveness in directing the generation of adversarial inputs. As the paper notes, there is no evidence that the metrics achieve better efficiency and efficacy than existing adversarial example generation methods. Additionally, further experiments in [62] suggest that existing coverage metrics might be ineffective for fault detection of DNNs with natural, non-adversarial inputs. This seems intuitive: minimal adversarial robustness is concerned with robustness on individual paths on a low level correlating well with structural coverage on neurons, while classifying "natural images" is concerned with global, functional behavior on a higher level, necessitating a different kind of coverage.

Instead of posing coverage on the original model, we can also consider coverage on a "proxy model" such as a variational autoencoder (VAE) that learns the input domain from data [18]. Such a proxy model may be useful due do the *manifold hypothesis*: relevant images do not fill the complete input space of possible images, but that natural images lie on an low-dimensional manifold. A proxy model tries to capture this manifold. Byun *et al.* [18] describe the idea of *k-section manifold combination coverage*. They use a VAE that is constructed to have a smaller latent space that still allows to capture the complete manifold in the input space. This latent space layer in the proxy model allows to formulate a structural coverage metric known from white-box coverage to the latent space of the VAE and thus the manifold in the input space. First results indicate that this may provide better feedback than structural coverage metrics such as neuron coverage [72].

Discussion. It is clear that on a low-level, data-flow oriented DNNs cannot be evaluated using traditional source code metrics [72]. First approaches to a neuron coverage showed that using a neuron definition may help in identifying fault revealing tests [72] and help measure the contribution of adversarial examples to a test set [65]. Although, the overall effectiveness of these metrics still needs to be shown, such metrics may help to identify issues with the network or the test set, *e.g.*, identifying that the test set may not completely activate the input layer [1]. The selection of a relevant structural coverage model is related to the kind of errors can and should be detected on a structural level. We discuss this further in the context of mutation testing below. While it is still unclear if structural coverage on a proxy model is a good metric for test adequacy, proxy model-based approaches are interesting for verification and validation as they have further applications, *e.g.*, for input generation [112], anomaly detection [36] and identifying missing data [101]. Another research direction is to shift from a structural level to a semantic level and extract semantic concepts from activations [55] and use those concepts for coverage, *e.g.*, with references to a semantic domain model (*cf.* Sec. 3.2.3).

5.2 Mutation Coverage

Mutation testing [6] uses the concept of a mutant, *i.e.* a SuT with a defined change (mutation), in order to evaluate the effectiveness of a test set. Mutation testing rests on the assumptions that (*i*) (small) mutations should correspond to actual mistakes programmers introduce and (*ii*) these small mistakes actually results in observable misbehavior. If a test can observe this misbehavior and identify a mutation, the test “kills” the mutant. The corresponding mutation coverage (score) computes how many of the generated mutants, can be killed by a test suite. While mutation testing faces several challenges such as the equivalent mutant problem and computational costs, it can be highly effective [6]. For DNNs different concepts of mutations are needed and several works have proposed different mutation operations for deep learning [28, 66, 92]. In contrast to program mutation, the described mutation operators perform changes on different parts of a deep learning and in different granularity: Mutation is applied to data, network architecture, as well as learned weights and biases. Jahangirova *et al.* [49] categorize proposed mutation operators and experimentally identify interesting mutation operators as well propose a new definition of “killing a mutant” that takes the stochastic nature of DNNs into account.

Discussion. Mutation testing is a useful concept for adequacy of a given test set *w.r.t.* the considered errors. However, it is important to define **relevant** errors. Previous work [49, 109] discusses that more research is needed to better design mutation operators. In particular, it is not clear whether currently defined mutation operators such as *Activation Function Removal* or *Layer Removal* relate to actual errors. Therefore further research is need to (*i*) evaluate already available mutation operators *w.r.t.* adequacy and (*ii*) use available sources of errors such as the taxonomy by Humbatova *et al.* [45]. An argumentation of the relevance of generated mutants strengthens the argumentation of mutation coverage of a test set.

5.3 Model-based input coverage

Input domain models, *e.g.*, based on semantic domain models as discussed in Sec. 3.2.3, help to structure the input space based on expert knowledge and intuition. They are intended to build equivalence classes for defining adequate test sets and to pave the way for a test end criterion. The high dimensionality of input domain models, however, still prohibits a full exploration in terms of test data. To this end, more sparse sampling techniques like combinatorial testing [59, 69] can be used to reduce the sampling space.

Gladisch *et al.* [42] describe a (partial) domain model for automated driving with 15 dimensions that specify a road network with road properties (such as *road-type*), and lane properties (such as *lane-type*). Each dimension can have two or more possible values resulting in a state space of $1.8 \cdot 10^{12}$ combinations. Pair-wise combinatorial test generation reduces the $1.8 \cdot 10^{12}$ combinations for the road network to 420 test cases. In another model for pedestrian appearance the full combinatorial test space ($4.2 \cdot 10^{22}$) can be reduced to 189 test cases satisfying pair-wise combinatorial coverage. How much the combinatorial space can be reduced significantly depends on the number of visual parameters and the number of different choices for each visual parameter. Gladisch *et al.* [42] use existing combinatorial coverage metrics and corresponding tools with a focus on collaborative and iterative domain model development and its management. Furthermore, existing datasets are analyzed and missing tests are generated using an iterative workflow.

Cheng *et al.* [20] focus on affordances provided by a simulator to generate synthetic data. They propose a new quantitative combinatorial testing metric which allows to define the minimum amount of samples per dimension, hence giving importance to dimensions. The paper focuses mainly on algorithms for satisfying the metric and discusses their computational complexity. On the use-case side they discuss how to cover a specified domain based on so-called scenarios using their metric and show its application in synthetic data generation using CARLA [25].

Discussion. Model-based input coverage based on semantic domain models (*cf.* Sec. 3.2.3) aims at structured testing and tries to connect to expert knowledge and experience in testing (non-ML) perception functions. Testing the function under controlled influence factor variations can help to avoid hidden stratification [71] as stratification is performed as a dedicated and reviewable task. A problem that remains *w.r.t.* model-based input coverage is the mapping between the (abstract) semantic domain model and concrete images.

In either direction the mapping is not unique: (*i*) Mapping abstract dimensions to concrete images is an open problem, because the visual appearance of phenomenological entities can be very different and, thus, many different realizations of a semantic concept exist and (*ii*) mapping concrete images to the semantic domain requires data annotations that are typically not available in public datasets and that, in some cases, require additional measurements during data collection. If the semantic domain models are very large, automating the labeling process is crucial. Combinatorial testing may help in dealing with the exponential growth in input interactions for such large models. While the effectiveness of combinatorial testing is established for domains such as application and embedded software [59, 69], its effectiveness for testing a computer vision task is yet to be shown. Therefore, it seems promising to combine input domain models to systematically generate images with specific contents with validated synthetic data generation as discussed above in Sec. 3.3.

6 COMBINING APPROACHES INTO TEST SET STRATEGIES

In this paper we posed the question how to test a machine learning-based perception function given incomplete specification and data. As the function is defined as an input-output relation, examples of which are given by the data and corresponding labels, standard software testing techniques need to be extended. Such perception functions are embedded into an overall automated system as shown in Fig. 1. For the different test scopes shown by the braces in the figure, techniques from different fields are relevant. While a smaller scope (1) leverages techniques from computer vision and machine learning, such as invariance under image augmentation (*cf.* Sec. 3.1), a larger scope (4) considers a closed-loop, cyber-physical system test scope, *e.g.*, using simulation (*cf.* Sec. 3.3.3) and deriving specifications from a system view as presented in Sec. 4.2.2.

We have separated the content into three parts guided by the following questions: *How can ML test engineers create relevant and meaningful test data efficiently for deep learning-based visual* *perception tasks? How can ML test engineers verify relevant properties of the corresponding DNNs? And how can ML test engineers show test adequacy of the available test data?*

As we can see from the corresponding discussions, test inputs and test oracles are highly dependent and cannot be separated. This is not surprising due to the data-driven nature of machine learning and the perception task. This leads to the necessity of various collections of test sets (not to be dismissed with a test dataset) where each test set has a specific **test purpose**. Each test set is based on a combination of input and oracle. Below, we describe several such purposes and how the corresponding test set is constructed based on a combination of the methods discussed above. Note that this notion of curating test sets with specific images and corresponding properties has been discussed before by practitioners in the field [54].

(*TS0*) **Standard ML test set:** The basic test set for evaluating performance as typically used in machine learning is a minimal requirement, since without performance evaluation, even the satisfaction of average case performance cannot be determined. There may even be several versions of such sets, *e.g.*, for development and later release stages.

(*TS1*) **Invariance in domain:** A second test set could show that DNN performance is invariant under relevant input variations. This approach leverages a semantic domain model (*cf.* Sec. 3.2.3) and uses a validated synthetic data generation (*cf.* Sec. 3.3) to provide data that covers the input domain. Here, we can check corresponding performance metrics (*cf.* Sec. 4.2.2) and verify performance invariance and check that there are no hidden stratifications (as discussed in Sec. 5.3.). A similar approach can also be used to derive a more accurate performance evaluation of the function by preventing averaging out worst cases.

(*TS2*) **DNN corner cases:** There will always be a need for a strong test set based on real data whether for safety argumentation or validation of a synthetic data pipeline. Here, we may use more fine-grained metrics for evaluation, *e.g.*, by considering relevance of pedestrians in pedestrian detection (*cf.* Fig. 5). For such real data, ground truth annotations are a bottleneck. Here active learning [90], supported by a semantic domain model (*cf.* Sec. 3.2.3) as well as coverage metrics (*cf.* Sec. 5.1), may help to select relevant new data for labeling.

(*TS3*) **System corner cases:** A particular concern for safety is the performance in rare cases [101], which could be approached with the following test set. Here, we leverage knowledge of behavioral corner cases in the ODD (*cf.* Sec. 3.2.1), simulate them in a closed-loop simulator and evaluate behavior of the DNN, *e.g.*, leveraging this setup for CPS-based falsification (*cf.* Sec. 3.3.3).

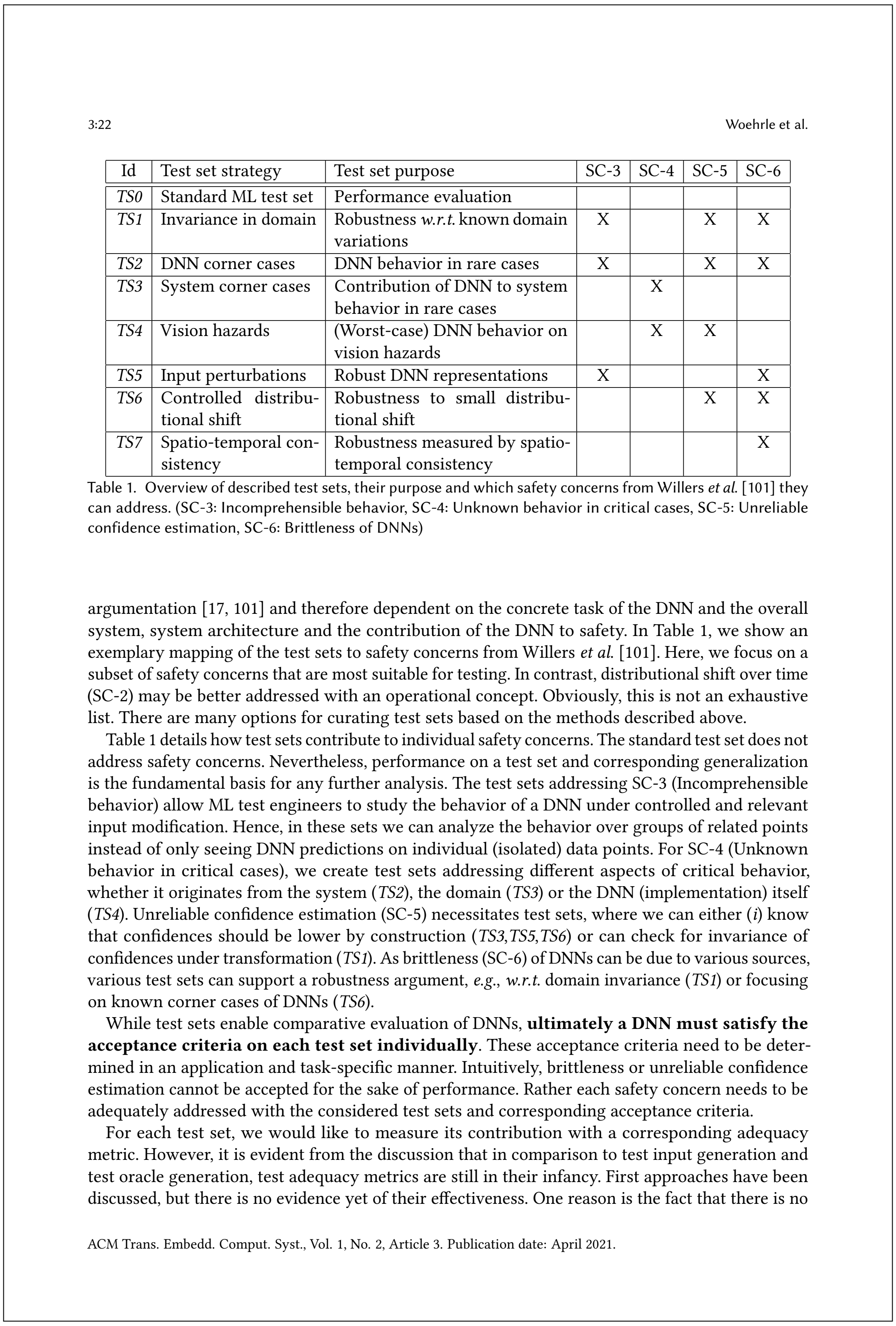
(*TS4*) **Vision hazards:** A similar test set could be based on known CV hazards annotated on real images, *e.g.*, based on CV HAZOP , and check for robustness to computer vision-specific corner cases. The WildDash [107] dataset is an example for such a test set. We detail on such hazard-based approaches in Sec. 3.2.2.

(*TS5*) **Input perturbations:** A further test set should consider robustness of the perception function *w.r.t.* input perturbations, *e.g.*, derived from the ODD (*cf.* Sec. 3.1.2), applied to real images. Corresponding properties are derived for each perturbation such as equivariance under translation and invariance under (partial) occlusion (*cf.* Sec. 4.3).

(*TS6*) **Controlled distributional shift:** TS5 uses modification of individual images for robustness testing. Another notion of robustness is to evaluate the effect of (small) data shifts without modification. This test set would use a related dataset with a data shift, *e.g.*, from a different camera, and check its effect on performance.

(*TS7*) **Spatio-temporal consistency:** A CPS such as an automated vehicle performs actions over time and space. Hence a test set based on sequences of real images, *e.g.*, from test drives, allows us to check spatio-temporal properties (*cf.* Sec. 4.2.1).

These test sets are complementary, yet not completely orthogonal and often offer synergies *w.r.t.* test input generation and test oracles. Test set purposes are closely aligned to the safety



Id	Test set strategy	Test set purpose	SC-3	SC-4	SC-5	SC-6
TS0	Standard ML test set	Performance evaluation				
TS1	Invariance in domain	Robustness *w.r.t.* known domain variations	X		X	X
TS2	DNN corner cases	DNN behavior in rare cases	X		X	X
TS3	System corner cases	Contribution of DNN to system behavior in rare cases		X		
TS4	Vision hazards	(Worst-case) DNN behavior on vision hazards		X	X	
TS5	Input perturbations	Robust DNN representations	X			X
TS6	Controlled distributional shift	Robustness to small distributional shift			X	X
TS7	Spatio-temporal consistency	Robustness measured by spatio-temporal consistency				X

Table 1. Overview of described test sets, their purpose and which safety concerns from Willers *et al.* [101] they can address. (SC-3: Incomprehensible behavior, SC-4: Unknown behavior in critical cases, SC-5: Unreliable confidence estimation, SC-6: Brittleness of DNNs)

argumentation [17, 101] and therefore dependent on the concrete task of the DNN and the overall system, system architecture and the contribution of the DNN to safety. In Table 1, we show an exemplary mapping of the test sets to safety concerns from Willers *et al.* [101]. Here, we focus on a subset of safety concerns that are most suitable for testing. In contrast, distributional shift over time (SC-2) may be better addressed with an operational concept. Obviously, this is not an exhaustive list. There are many options for curating test sets based on the methods described above.

Table 1 details how test sets contribute to individual safety concerns. The standard test set does not address safety concerns. Nevertheless, performance on a test set and corresponding generalization is the fundamental basis for any further analysis. The test sets addressing SC-3 (Incomprehensible behavior) allow ML test engineers to study the behavior of a DNN under controlled and relevant input modification. Hence, in these sets we can analyze the behavior over groups of related points instead of only seeing DNN predictions on individual (isolated) data points. For SC-4 (Unknown behavior in critical cases), we create test sets addressing different aspects of critical behavior, whether it originates from the system (*TS2*), the domain (*TS3*) or the DNN (implementation) itself (*TS4*). Unreliable confidence estimation (SC-5) necessitates test sets, where we can either (*i*) know that confidences should be lower by construction (*TS3*,*TS5*,*TS6*) or can check for invariance of confidences under transformation (*TS1*). As brittleness (SC-6) of DNNs can be due to various sources, various test sets can support a robustness argument, *e.g.*, *w.r.t.* domain invariance (*TS1*) or focusing on known corner cases of DNNs (*TS6*).

While test sets enable comparative evaluation of DNNs, **ultimately a DNN must satisfy the acceptance criteria on each test set individually**. These acceptance criteria need to be determined in an application and task-specific manner. Intuitively, brittleness or unreliable confidence estimation cannot be accepted for the sake of performance. Rather each safety concern needs to be adequately addressed with the considered test sets and corresponding acceptance criteria.

For each test set, we would like to measure its contribution with a corresponding adequacy metric. However, it is evident from the discussion that in comparison to test input generation and test oracle generation, test adequacy metrics are still in their infancy. First approaches have been discussed, but there is no evidence yet of their effectiveness. One reason is the fact that there is no commonly agreed evaluation approach yet. Additionally, further studies are needed how coverage can be leveraged to argue test set completeness, *e.g.*, by using a form of input domain coverage and complementing it with an error analysis.

7 CONCLUSION

This work reviews and discusses existing work on testing DNNs for visual perception in the context of automated driving. We separate the presentation into three parts for (*i*) test input generation, (*ii*) test oracle generation and (*iii*) test adequacy. We finally show how to combine the approaches into test set strategies. As pointed out by [62, 109], there is a need for more detailed and comparable evaluation of test methods in order to actually determine efficiency and effectiveness of individual methods. We see the need for considering a more diverse discussion of testing, *e.g.*, different notions of robustness testing, task-specific error models, test-specific metrics and different notions of neural network coverage, similar to the research opportunities in ML testing discussed in [109]. This work tries to show that interaction is needed between different fields such as cyber-physical systems, computer vision, machine learning and software testing such that individual methods from different domains can be joined into efficient and effective verification approaches.

8 ACKNOWLEDGEMENTS

The research leading to the results presented above are funded by the German Federal Ministry for Economic Affairs and Energy within the project *KI Absicherung – Safe AI for automated driving*.

REFERENCES

[1] Stephanie Abrecht, Maram Akila, Sujan Sai Gannamaneni, Konrad Groh, Christian Heinzemann, Sebastian Houben, and Matthias Woehrle. 2020. Revisiting Neuron Coverage and its Application to Test Generation. In *Computer Safety, Reliability, and Security. SAFECOMP Workshops*. Springer.

[2] National Highway Traffic Safety Administration et al. 2017. Automated driving systems 2.0: A vision for safety. *Washington, DC: US Department of Transportation, DOT HS* 812 (2017), 442.

[3] AIT Austrian Institute of Technology GmbH. 2019. CV-HAZOP VITRO. (Mar 2019). Retrieved March 28, 20019 from https://vitro-testing.com/cv-hazop/

[4] Naveed Akhtar and Ajmal Mian. 2018. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* 6 (2018), 14410–14430.

[5] Hassan Abu Alhaija, Siva Karthik Mustikovela, Lars M. Mescheder, Andreas Geiger, and Carsten Rother. 2018. Augmented Reality Meets Computer Vision: Efficient Data Generation for Urban Driving Scenes. *Journal of Computer Vision* 126, 9 (2018), 961–972.

[6] Paul Ammann and Jeff Offutt. 2016. *Introduction to software testing.* Cambridge University Press.

[7] ASAM. 2020. *P2019-09 OpenLABEL - Project Proposal.* https://www.asam.net/project-detail/scenario-storage-and-labelling/

[8] Gerrit Bagschik, Till Menzel, and Markus Maurer. 2018. Ontology based Scene Creation for the Development of Automated Vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV).* 1813–1820.

[9] Anand Balakrishnan, Aniruddh G Puranic, Xin Qin, Adel Dokhanchi, Jyotirmoy V Deshmukh, Heni Ben Amor, and Georgios Fainekos. 2019. Specifying and Evaluating Quality Metrics for Vision-based Perception Systems. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE).* IEEE, 1433–1438.

[10] Mayank Bansal, Alex Krizhevsky, and Abhijit S. Ogale. 2019. ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst. In *Robotics: Science and Systems XV.*

[11] Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. 2015. The Oracle Problem in Software Testing: A Survey. *IEEE Trans. Software Eng.* 41, 5 (2015), 507–525.

[12] Jan-Aike Bolte, Andreas Bar, Daniel Lipinski, and Tim Fingscheidt. 2019. Towards corner case detection for autonomous driving. In *2019 IEEE Intelligent Vehicles Symposium (IV).* IEEE, 438–445.

[13] Markus Borg, Cristofer Englund, Krzysztof Wnuk, Boris Durann, Christoffer Lewandowski, Shenjian Gao, Yanwen Tan, Henrik Kaijser, Henrik Lönn, and Jonnas Törnqvist. 2019. Safely Entering the Deep : A Review of Verification and Validation for Machine Learning and a Challenge Elicitation in the Automotive Industry. *Journal of Automotive Software Engineering* 1, 1 (2019), 1–19.

[14] Tom Boyd. 2020. Rethinking Cruise's AV Development Loop During COVID-19. (May 2020). Retrieved May 26, 2020 from https://medium.com/cruise/cruise-av-development-loop-covid-19-1daef2f0c3d5

[15] Houssem Ben Braiek and Foutse Khomh. 2020. On testing machine learning programs. *Journal of Systems and Software* 164 (2020), 110542.

[16] Simon Burton, Lydia Gauerhof, and Christian Heinzemann. 2017. Making the Case for Safety of Machine Learning in Highly Automated Driving. In *Computer Safety, Reliability, and Security*. Springer, 5–16.

[17] Simon Burton, Lydia Gauerhof, Bibhuti Bhusan Sethy, Ibrahim Habli, and Richard Hawkins. 2019. Confidence Arguments for Evidence of Performance in Machine Learning for Highly Automated Driving Functions. In *Computer Safety, Reliability, and Security*. Springer.

[18] Taejoon Byun and Sanjai Rayadurgam. 2020. Manifold for Machine Learning Assurance. In *Proc. Int'l Conf. on Software Engineering: New Ideas and Emerging Results*.

[19] Alexandra Carlson, Katherine A. Skinner, Ram Vasudevan, and Matthew Johnson-Roberson. 2019. Sensor Transfer: Learning Optimal Sensor Effect Image Augmentation for Sim-to-Real Domain Adaptation. *IEEE Robotics and Automation Letters* 4, 3 (2019), 2431–2438. https://doi.org/10.1109/LRA.2019.2896470

[20] Chih-Hong Cheng, Chung-Hao Huang, and Hirotoshi Yasuoka. 2018. Quantitative projection coverage for testing ML-enabled autonomous systems. In *Int'l Symposium on Automated Technology for Verification and Analysis*. Springer, 126–142.

[21] Shuyang Cheng, Yang Song, and Peisheng Li. 2020. Using automated data augmentation to advance our Waymo Driver. (Apr 2020). Retrieved Nov. 2020 from https://blog.waymo.com/2020/04/using-automated-data-augmentation-to.html

[22] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. 2016. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *CVPR 2016*.

[23] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. 2020. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR 2020 Workshops*. 702–703.

[24] Martin D Davis and Elaine J Weyuker. 1981. Pseudo-oracles for non-testable programs. In *Proc. of the ACM'81 Conference*. 254–257.

[25] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proc. Annual Conf. on Robot Learning*. 1–16.

[26] Tommaso Dreossi, Alexandre Donzé, and Sanjit A Seshia. 2019. Compositional falsification of cyber-physical systems with machine learning components. *Journal of Automated Reasoning* 63, 4 (2019), 1031–1053.

[27] Tommaso Dreossi, Shromona Ghosh, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. 2019. A Formalization of Robustness for Deep Neural Networks. In *Proc. AAAI 2019 Spring Symposium, Verification of Neural Networks*.

[28] Anurag Dwarakanath, Manish Ahuja, Samarth Sikand, Raghotham M. Rao, R. P. Jagadeesh Chandra Bose, Neville Dubash, and Sanjay Podder. 2018. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. In *Proc. 27th ACM International Symposium on Software Testing and Analysis*. 118–128.

[29] Hala Elforia, Jan-Pieter Paardekooper, Erwin de Gelder, Sytze Kalisvaart, and Olaf Op den Camp. 2018. *Streetwise - Scenario-based Safety Validation of Connected and Automated Driving*. White Paper. TNO.

[30] Marius Feilhauer, Juergen Haering, and Sean Wyatt. 2016. Current approaches in HiL-based ADAS testing. *SAE Int'l Journal of Commercial Vehicles* 9, 2016-01-8013 (2016), 63–69.

[31] Association for Standardization of Automation and Measuring Systems e.V. 2020. ASAM OpenScenario. (2020). Retrieved November 4, 2020 from https://www.asam.net/standards/detail/openscenario/

[32] Uwe Franke. 2018. 30 Years Fighting for Robustness. (June 2018). Retrieved May 29, 2020 from https://www.youtube.com/watch?v=AOP8iitFbPY Talk at Robust Vision Challenge 2018.

[33] Daniel J. Fremont, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. 2019. Scenic: A Language for Scenario Specification and Scene Generation. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2019)*. ACM, 63–78.

[34] Benoît Frénay and Ata Kabán. 2014. A comprehensive introduction to label noise. In *ESANN 2014*.

[35] Jelena Frtunikj and Simon Fuerst. 2019. Engineering Safe Machine Learning for Automated Driving Systems. In *27th Safety-Critical Systems Symposium*.

[36] Lydia Gauerhof and Nianlong Gu. 2020. Reverse Variational Autoencoder for Visual Attribute Manipulation and Anomaly Detection. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2103–2112.

[37] Lydia Gauerhof, Richard Hawkins, Chiara Picardi, Colin Paterson, Yuki Hagiwara, and Ibrahim Habli. 2020. Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings. In *Computer Safety, Reliability, and Security*. Springer.

[38] Lydia Gauerhof, Peter Munk, and Simon Burton. 2018. Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving. In *Computer Safety, Reliability, and Security*. Springer.

[39] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh

Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. 2020. A2D2: Audi Autonomous Driving Dataset. (2020). arXiv:cs.CV/2004.06320 https://www.a2d2.audi

[40] GIDAS. 2020. German In-Depth Accident Study (GIDAS). (2020). Retrieved Nov., 2020 from https://www.gidas.org

[41] Christoph Gladisch, Thomas Heinz, Christian Heinzemann, Jens Oehlerking, Anne von Vietinghoff, and Tim Pfitzer. 2019. Experience Paper: Search-Based Testing in Automated Driving Control Applications. In *34th IEEE/ACM Int'l Conf. on Automated Software Engineering (ASE)*. 26–37.

[42] Christoph Gladisch, Christian Heinzemann, Martin Herrmann, and Matthias Woehrle. 2020. Leveraging combinatorial testing for safety-critical computer vision datasets. In *Workshop on Safe Artificial Intelligence for Automated Driving.*

[43] Thomas R. Gruber. 1995. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies* 43, 5 (1995), 907 – 928.

[44] Dan Hendrycks and Thomas G. Dietterich. 2019. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *ICLR 2019.*

[45] Nargiz Humbatova, Gunel Jahangirova, Gabriele Bavota, Vincenzo Riccio, Andrea Stocco, and Paolo Tonella. 2020. Taxonomy of Real Faults in Deep Learning Systems. In *Proc. 42nd International Conference on Software Engineering.*

[46] Alexander Hutter. 2008. Einsatz von Simulationsmodellen beim Test elektronischer Steuergeräte. In *Automatisiertes Testen Eingebetteter Systeme in der Automobilindustrie*, Eric Sax (Ed.). Hanser.

[47] ISO. 2018. *Road vehicles – Functional safety.* ISO ISO 26262-(1-12):2018. International Organization for Standardization, Geneva, Switzerland.

[48] ISO. 2019. *Road vehicles – Safety of the intended functionality.* ISO ISO/PAS 21448. International Organization for Standardization, Geneva, Switzerland.

[49] Gunel Jahangirova and Paolo Tonella. 2020. An Empirical Evaluation of Mutation Operators for Deep Learning Systems. In *IEEE Int'l Conf. on Software Testing, Verification and Validation (ICST).*

[50] Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. 2020. Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art. *Found. Trends Comput. Graph. Vis.* 12, 1-3 (2020), 1–308.

[51] Susmit Jha, Vasumathi Raman, Dorsa Sadigh, and Sanjit A Seshia. 2018. Safe autonomy under perception uncertainty using chance-constrained temporal logic. *Journal of Automated Reasoning* 60, 1 (2018), 43–62.

[52] Arut Prakash Kaleeswaran, Arne Nordmann, Thomas Vogel, and Lars Grunske. 2020. Counterexample Interpretation for Contract-Based Design. In *Int'l Symp. on Model-Based Safety and Assessment.* Springer, 99–114.

[53] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. 2019. Meta-sim: Learning to generate synthetic datasets. In *Proceedings of the IEEE International Conference on Computer Vision.* 4551–4560.

[54] Andrej Karpathy. 2020. AI for Full-Self Driving. (Apr 2020). Retrieved June 5, 2020 from http://scaledml.org/2020/ Presentation 5th Scaled Machine Learning Conf.

[55] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. 2018. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *Proc. 35th Int'l Conf. on Machine Learning, ICML.* PMLR, 2673–2682.

[56] Kodiak. 2020. Kodiak Safety Report. (2020). Retrieved June 15, 2020 from https://kodiak.ai/safety-report/

[57] Philip Koopman and Frank Fratrik. 2019. How Many Operational Design Domains, Objects, and Events?. In *Workshop on AI Safety @ AAAI 2019.*

[58] Philip Koopman and Michael Wagner. 2018. *Toward a framework for highly automated vehicle safety validation.* Technical Report. SAE Technical Paper.

[59] D. R. Kuhn, D. R. Wallace, and A. M. Gallo. 2004. Software fault interactions and implications for software testing. *IEEE Transactions on Software Engineering* 30, 6 (June 2004), 418–421.

[60] Ulrich Lages, Martin Spencer, and Roman Katz. 2013. Automatic scenario generation based on laserscanner reference data and advanced offline processing. In *2013 IEEE Intelligent Vehicles Symposium (IV).* IEEE, 153–155.

[61] Andreas Leitner, Ilinca Ciupa, Manuel Oriol, Bertrand Meyer, and Arno Fiva. 2007. Contract driven development= test driven development-writing test cases. In *Proc. 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering.* 425–434.

[62] Zenan Li, Xiaoxing Ma, Chang Xu, and Chun Cao. 2019. Structural coverage criteria for neural networks could be misleading. In *Proc. Int'l Conf. on on Software Engineering: New Ideas and Emerging Results.* 89–92.

[63] Changliu Liu, Tomer Arnon, Christopher Lazarus, Clark Barrett, and Mykel J. Kochenderfer. 2019. Algorithms for Verifying Deep Neural Networks. *CoRR* abs/1903.06758 (2019). arXiv:1903.06758

[64] Albert Lutz, Bernhard Schick, Henning Holzmann, Michael Kochem, Harald Meyer-Tuve, Olav Lange, Yiqin Mao, and Guido Tosolin. 2017. Simulation methods supporting homologation of Electronic Stability Control in vehicle variants. *Vehicle system dynamics* 55, 10 (2017), 1432–1497.

[65] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: multi-granularity testing criteria for deep learning systems. In

Proc. ACM/IEEE Conf. on Automated Software Engineering (ASE). 120–131.

[66] Lei Ma, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Felix Juefei-Xu, Chao Xie, Li Li, Yang Liu, Jianjun Zhao, et al. 2018. Deepmutation: Mutation testing of deep learning systems. In *Proc. Int'l Symp on Software Reliability Engineering (ISSRE).* IEEE, 100–111.

[67] Nikolaus Mayer, Eddy Ilg, Philipp Fischer, Caner Hazirbas, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. 2018. What Makes Good Synthetic Training Data for Learning Disparity and Optical Flow Estimation? *Int'l Journal of Computer Vision* 126, 9 (2018), 942–960.

[68] Betrand Meyer. 2019. Soundness and completeness: with precision. (Apr 2019). Retrieved May 28, 2020 from https://bertrandmeyer.com/2019/04/21/soundness-completeness-precision

[69] NIST. 2020. Automated Combinatorial Testing for Software. (2020). https://csrc.nist.gov/projects/automated-combinatorial-testing-for-software Accessed 2020-05-29.

[70] Farzan Erlik Nowruzi, Prince Kapoor, Dhanvin Kolhatkar, Fahed Al Hassanat, Robert Laganiere, and Julien Rebut. 2019. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. In *International Conference on Machine Learning (ICML) Workshops.*

[71] Luke Oakden-Rayner, Jared Dunnmon, Gustavo Carneiro, and Christopher Re. 2020. Hidden Stratification Causes Clinically Meaningful Failures in Machine Learning for Medical Imaging. In *Proc. of the ACM Conference on Health, Inference, and Learning.* ACM, 151–159.

[72] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In *Proc. 26th Symposium on Operating Systems Principles (SOSP).* 1–18.

[73] Zachary Pezzementi, Trenton Tabor, Samuel Yim, Jonathan K Chang, Bill Drozd, David Guttendorf, Michael Wagner, and Philip Koopman. 2018. Putting image manipulations in context: robustness testing for safe perception. In *2018 IEEE Int'l Symp. on Safety, Security, and Rescue Robotics (SSRR).* 1–8.

[74] Atanas Poibrenski, Janis Sprenger, and Christian Muller. 2018. Toward a Methodology for Training with Synthetic Data on the Example of Pedestrian Detection in a Frame-by-Frame Semantic Segmentation Task. In *SEFAIAS@ICSE 2018.* 31–34.

[75] Chongli Qin, Krishnamurthy (Dj) Dvijotham, Brendan O'Donoghue, Rudy Bunel, Robert Stanforth, Sven Gowal, Jonathan Uesato, Grzegorz Swirszcz, and Pushmeet Kohli. 2019. Verification of Non-Linear Specifications for Neural Networks. In *ICLR 2019.*

[76] Christopher Ré. 2018. Software 2.0 and Snorkel: Beyond Hand-Labeled Data. In *Proc. 24th ACM KDD 2018, August 19-23, 2018.* 2876.

[77] German Ros, Vladlen Koltun, Felipe Codevilla, and Antonio M. Lopez. 2020. CARLA Autonomous Driving Challenge. (2020). Retrieved May 29, 2020 from https://carlachallenge.org/

[78] Francisca Rosique, Pedro J. Navarro, Carlos FernÃąndez, and Antonio Padilla. 2019. A Systematic Review of Perception System and Simulators for Autonomous Vehicles Research. *Sensors* 19, 3 (2019). https://doi.org/10.3390/s19030648

[79] SAE 2018. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles.* SAE. https://doi.org/10.4271/J3016_201806

[80] Selma Saidi, Sebastian Steinhorst, Arne Hamann, Dirk Ziegenbein, and Marko Wolf. 2018. Future automotive systems design: research challenges and opportunities: special session. In *Proc. Int'l Conf. on Hardware/Software Codesign and System Synthesis.* IEEE Press, 2.

[81] Rick Salay and Krzysztof Czarnecki. 2019. Improving ML Safety with Partial Specifications. In *Computer Safety, Reliability, and Security.* Springer, 288–300.

[82] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. 2017. An analysis of ISO 26262: Using machine learning safely in automotive software. *arXiv preprint arXiv:1709.02435* (2017).

[83] Alberto Sangiovanni-Vincentelli, Werner Damm, and Roberto Passerone. 2012. Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. *European journal of control* 18, 3 (2012), 217–238.

[84] Gesina Schwalbe and Martin Schels. 2020. A Survey on Methods for the Safety Assurance of Machine Learning Based Systems. In *10th European Congress on Embedded Real Time Software and Systems (ERTS 2020).*

[85] SCSC. 2019. *Data Safety Guidance.* SCSC Version 3.1. The Safety-Critical Systems Club, York, Great Britain.

[86] D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, and Dan Dennison. 2015. Hidden technical debt in machine learning systems. In *NeurIPS 2015.* 2503–2511.

[87] Sergio Segura, Gordon Fraser, Ana B. Sánchez, and Antonio Ruiz Cortés. 2016. A Survey on Metamorphic Testing. *IEEE Trans. Software Eng.* 42, 9 (2016), 805–824.

[88] Koushik Sen, Darko Marinov, and Gul Agha. 2005. CUTE: a concolic unit testing engine for C. *ACM SIGSOFT Software Engineering Notes* 30, 5 (2005), 263–272.

[89] Sanjit A. Seshia, Ankush Desai, Tommaso Dreossi, Daniel J. Fremont, Shromona Ghosh, Edward Kim, Sumukh Shivakumar, Marcell Vazquez-Chanlatte, and Xiangyu Yue. 2018. Formal Specification for Deep Neural Networks. In

ATVA 2018. 20–34.

[90] Burr Settles. 2011. From theories to queries: Active learning in practice. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*. 1–18.

[91] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2018. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*. Springer, 621–635.

[92] Weijun Shen, Jun Wan, and Zhenyu Chen. 2018. Munn: Mutation analysis of neural networks. In *2018 IEEE Int'l Conf. on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 108–115.

[93] Rakshith Shetty, Bernt Schiele, and Mario Fritz. 2019. Not Using the Car to See the Sidewalk–Quantifying and Controlling the Effects of Context in Classification and Segmentation. In *CVPR 2019*. 8218–8226.

[94] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. 2019. An Abstract Domain for Certifying Neural Networks. *Proc. ACM Program. Lang.* 3, POPL, Article 41 (Jan. 2019), 30 pages. https://doi.org/10.1145/3290354

[95] Youcheng Sun, Min Wu, Wenjie Ruan, Xiaowei Huang, Marta Kwiatkowska, and Daniel Kroening. 2018. Concolic testing for deep neural networks. In *Proc. ACM/IEEE Conf. on Automated Software Engineering, (ASE)*. 109–119.

[96] Y. Tian, X. Li, K. Wang, and F. Wang. 2018. Training and testing object detectors with virtual images. *IEEE/CAA Journal of Automatica Sinica* 5, 2 (2018), 539–546.

[97] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*. 303–314.

[98] Raquel Urtasun. 2018. A future with affordable self-driving vehicles. (Dec 2018). Retrieved May 29, 2020 from https://wimlworkshop.org/2018/program/ Presentation at WiML.

[99] Serin Varghese, Yasin Bayzidi, Andreas Bar, Nikhil Kapoor, Sounak Lahiri, Jan David Schneider, Nico M Schmidt, Peter Schlicht, Fabian Huger, and Tim Fingscheidt. 2020. Unsupervised Temporal Consistency Metric for Video Segmentation in Highly-Automated Driving. In *CVPR 2020 Workshops*. 336–337.

[100] Nick Webb, Dan Smith, Christopher Ludwick, Trent Victor, Qi Hommes, Francesca Favaro, George Ivanov, and Tom Daniel. 2020. Waymo's Safety Methodologies and Safety Readiness Determinations. *arXiv preprint arXiv:2011.00054* (2020).

[101] Oliver Willers, Sebastian Sudholt, Shervin Raafatnia, and Stephanie Abrecht. 2020. Safety Concerns and Mitigation Approaches Regarding the Use of Deep Learning in Safety-Critical Perception Tasks. In *Computer Safety, Reliability, and Security. SAFECOMP Workshops*. Springer, 336–350.

[102] Matthias Woehrle, Christoph Gladisch, and Christian Heinzemann. 2019. Open Questions in Testing of Learned Computer Vision Functions for Automated Driving. In *Computer Safety, Reliability, and Security. SAFECOMP Workshops*. Springer, 333–345.

[103] Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. 2018. Scaling provable adversarial defenses. In *NeurIPS 2018*. 8410–8419.

[104] Magnus Wrenninge and Jonas Unger. 2018. Synscapes: A Photorealistic Synthetic Dataset for Street Scene Parsing. (2018). arXiv:cs.CV/1810.08705

[105] Alan L Yuille and Chenxi Liu. 2018. Deep Nets: What have they ever done for Vision? *arXiv:1805.04025* (2018).

[106] Oliver Zendel, Katrin Honauer, Markus Murschitz, Martin Humenberger, and Gustavo Fernández Domínguez. 2017. Analyzing Computer Vision Data - The Good, the Bad and the Ugly. In *CVPR 2017*. 6670–6680.

[107] Oliver Zendel, Katrin Honauer, Markus Murschitz, Daniel Steininger, and Gustavo Fernández Domínguez. 2018. WildDash - Creating Hazard-Aware Benchmarks. In *ECCV 2018*. 407–421.

[108] Oliver Zendel, Markus Murschitz, Martin Humenberger, and Wolfgang Herzner. 2015. CV-HAZOP: Introducing Test Data Validation for Computer Vision. In *Int'l Conf. on Computer Vision (ICCV)*. 2066–2074.

[109] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* (2020).

[110] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 2018. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In *Proc. ACM/IEEE Int'l Conf. on Automated Software Engineering*. 132–142.

[111] Yi Zhang, Weichao Qiu, Qi Chen, Xiaolin Hu, and Alan L. Yuille. 2018. UnrealStereo: Controlling Hazardous Factors to Analyze Stereo Vision. In *Proc. 3DV 2018*. 228–237.

[112] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. 2016. Generative visual manipulation on the natural image manifold. In *European Conf. on Computer Vision*. Springer, 597–613.

[113] Adrian Zlocki, Andreas Pütz, Julian Bock, and Lutz Eckstein. 2017. System validation of highly automated vehicles with a database of relevant traffic scenarios. In *12th ITS European Congress*.

[114] Zoox. 2020. Zoox: 1-Hour Fully Autonomous Drive in San Francisco with Commentary. (April 2020). Retrieved November 09, 2020 from https://youtu.be/6r7vDhPXmiM?t=601

On the Necessity of Explicit Artifact Links in Safety Assurance Cases for Machine Learning

Lydia Gauerhof, Roman Gansch, Christian Heinzemann, Matthias Woehrle, Andreas Heyl
Corporate Research
Robert Bosch GmbH
Renningen, Germany
[firstname.surname]@de.bosch.com

***Abstract*—The perception in autonomous systems is essential for safe behavior. Machine learning (ML)-based functions play an increasingly important role in this context. The development and safety assurance of such functions is different from the development of non-ML-based functions. Traceability of the various artifacts generated for safety argumentation is challenging, as there is i.e. no longer a direct mapping from requirements to code and data cannot be directly mapped to a semantic domain model. In this work, we show that and how the links between artifacts, which are created in different stages of the development, must be established explicitly. These links enable us to build confidence in our safety argumentation. We concretize these explicit links in two examples, namely pedestrian detection and vehicle detection.**

***Index Terms*—Machine Learning, safety, traceability.**

I. Introduction

The failure of autonomous safety-critical systems, such as self-driving vehicles, can result in fatalities [1]. As perception is essential for further system decisions regarding behavior and trajectory planning, it is necessary to perform a rigorous safety analysis for perception components, which are often implemented based on Machine Learning (ML), to establish a systematically justified safety assurance case (SAC) [2].

As most safety standards focus on a general and not ML-specific safety aspect, such as SOTIF [3] or ISO26262 [4], the regulatory structure that utilizes the artifacts created during the development for SACs for ML-based components is not distinctive. Furthermore, it is noticeable that in particular the artifacts from the system engineering and from different phases of the ML lifecycle, *requirements elicitation, data management, design/training, testing, and deployment*, shown in Figure 1, are often created and maintained by different teams. The teams working on the different phases have different expert knowledge and split responsibilities. For example, safety experts elaborate the component specification while ML experts elaborate the DNN component. While some exchange of artifacts takes place in the collaboration, for example, the requirements created by the safety experts are passed on to the ML experts or test experts, a consistent exchange and alignment of knowledge might not take place in such an extent that traceability can be guaranteed.

The research leading to the results presented above are funded by the German Federal Ministry for Economic Affairs and Energy within the project *KI Absicherung – Safe AI for automated driving*.

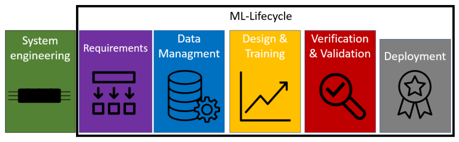


Fig. 1. We focus on the artifacts generated iterative and in parallel by system engineering and the phases of the ML-lifecycle [5], [6] requirement elicitation, data management, design&training, V&V. Deployment is out of scope.

For a 'traditional' SAC, it is necessary to trace requirements down to the code. However, a machine learning component is not explicitly programmed, but trained using data, and consequently there are no lines of code that can be traced back to a specific requirement. Adding to the black-box nature of the ML component, the open context in automated driving is complex and the potential input space represented by the data is very large. At the same time, data should be proven to be suitable for learning the intended functionality. Therefore, we must strengthen the links of artifacts from different phases of the development to enable a valid decomposition of the safety arguments and evidences to justify a high level of confidence in the safety of an ML component.

In this paper, we identify essential artifacts created in the different phases of the ML-lifecycle. Furthermore, we explain how they interdepent on each other and how traceability might be facilitated. We provide approaches how to realize and strengthen the concrete links between the artifacts. Our leading examples are two perception functions, DNN based pedestrian and vehicle detection functions, in an automated vehicle. The first case study deals with evaluating pedestrian detection with task-oriented metrics [7]. The second case study addresses the systematic modeling for environmental perception limitations in automated driving assessment [8]. We show how the links between artifacts manifest based on these two perception functions from the automated driving domain. We demonstrate which processes and methods are used to realize the linkage of particular aspects of the artifacts. In this way, we make aware how certain methods contribute to increase the confidence in the SAC of an ML component and which weaknesses remain.

II. Related Work

A SAC contains goal-based safety arguments [9], which in turn are decomposed into further arguments by a strategy. Evidence such as test results, development processes, or simulations support these arguments [10]. Established graphical representation of SACs are the Goal Structuring Notation (GSN) [11] and the Claims-Arguments-Evidence (CAE) Notation [12]. Although various SAC approaches for ML components are proposed [13]–[15], issues related to the artifact links are not yet sufficiently discussed.

Graphical notations, such as GSN and CAE, focus on the structuring of safety arguments, but they do not provide an approach to resolve the deeper technical challenges. In the development of safety-critical systems, a number of approaches are already used to closely interlink the individual development steps and to track the consequences of changes. However, there is no established approach to trace data and how to compensate the missing traces to the lines of code.

Requirement management tools, e.g. IBM Rational Doors, provide the functionality to trace requirements in order to be able to perform an impact assessment of changes to requirements. However, traceability of requirements for an ML is a challenge in itself. This is addressed in [5] by providing an approach that breaks down the requirements for an ML component from the system level via ML-specific desired properties, so-called desiderata. But requirements traceability alone is not sufficient as it might be the case, e.g., that the links to artifacts from data management are missing.

In software development, repositories are used for change management to make changes to code traceable and to facilitate merging of individual code fragments. However, change management does not solve the question of the integrity of the various artifacts in the SAC.

SafetyOps [16] is a concept to enable an efficient, continuous and traceable system safety life cycle. For this purpose, different automation frameworks (e.g., DevOps, TestOps, DataOps) are combined. However, the links of the individual frameworks are not explicitly elaborated.

The Guidance on the Assurance of Machine Learning in Autonomous Systems (AMLAS)" [15] fosters the links between artifacts for each stage through so-called Argument Patterns. These patterns provide an explanation on how the generated evidence supports the relevant safety claims of each stage. Although linking artifacts of different stages is seen as part of the iterations, it is not further discussed.

III. Artifacts

Before discussing the necessary explicit links, we describe the artifacts generated during development. We cluster them according to the lifecycle: system engineering, requirements, data management, design & training and V & V (see figure 1).

A. System engineering

• *Artifact for Operational Design Domain (ODD):* According to the standard ISO 34503 [17] on ODD Taxonomy, the ODD comprises the static and dynamic attributes within which an automated driving systems is designed to operate safely. Further, it excludes explicit conditions to restrict the open-world context [17].

• *Artifact for System Architecture:* The system architecture specifies the components that shall fulfill the intended functionality [3] and satisfy the safety goals within the ODD. The architecture of the perception subsystem where ML components are most relevant might be used, if the overall system architecture is not known. This contains, besides the ML component, sensors, preprocessing and components for fusing multiple perception channels.

B. Requirements

• *Artifact for Requirements:* In general, functional requirements and functional safety requirements are derived. Due to our focus, we refer here to functional safety requirements.

• *Artifact for Data Requirements:* The data requirements specify the data and define the required data properties.

• *Artifact for Data Acquisition Specification:* For data acquisition from the real world, it defines the spatial, geographic, and environmental factors (e.g. country, time of day, weather conditions), their frequency, their annotation to data and the configuration (e.g. camera model, installation location). For synthetically generated data, it defines in addition how the data shall be generated.

• *Artifact for labeling specification:* The labeling specification includes examples and determines the labeling quality [18].

• *Artifact for Semantic Domain Model (SDM):* The model of the input space, also called *semantic domain model*, is an extension of the ODD including the perception-specific properties [19]. In particular, the properties of sensors [20] and perception algorithms are taken into account due to their influence on data. The data features are disentagled from these and included seperately in the semantic domain model.

C. Data management

• *Artifact for Processed Data:* The data used for implementation of a ML component has to reflect the intended functionality that is expressed by the requirements. It is split into the development datasets and the V & V datasets based on the data split log. The development datasets consists of the training and validation dataset and the V & V datasets consists of the test dataset and supplemental data (e.g. out-of-distribution data). ML components often require multiple development iterations, and as consequence the used data is evolving. The data split log is essential to preserve the statistical inference validity [21].

• *Artifact for Labeling:* The label semantics and the labeling process are documented in this artifact.

• *Artifact for Annotations & Extracted Information:* Annotations of meta-information, such as weather conditions are contained in a separate artifact. It may also contain further extracted information which is e.g. obtained by transfering the data into a low-dimensional, latent space.

• *Data Analysis:* Extracted and annotated information are analyzed, w.r.t. distributions and correlations [22], [23].

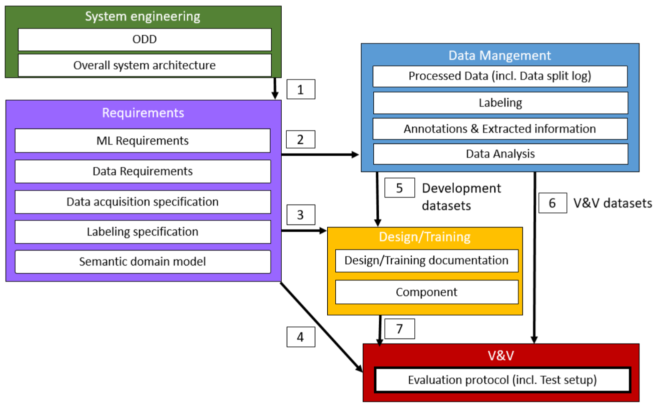


Fig. 2. Overview of artificats and links in a typical ML development process. The links are unidirectional to indicate the order of artifact creation. During development these links are instantiated bidirectional to obtain traceability. They should be made explicit, to avoid becoming contradictory or not sufficiently established. Consequently, the links allow the traceability of artifacts within an iterative development to gain confidence in the SAC and demonstrate validity.

D. Design & Training

• *Artifact for Design & Training Documentation:* For the design/training phase the algorithm, its specific architecture and the training parameters are chosen and documented in this artifact.

• *Artifact for Component:* For a DNN, the component includes the architecture and the trained weights.

E. Verification & Validation

For safety-critical functions, a statistical evaluation of average performance based on a data split is not sufficient, since testing must serve several purposes. In particular the evaluation of worst-case behavior and the search for critical errors and failures [19].

• *Artifact for Evaluation:* This artifact contains test setups, test results and the corresponding evaluations. During testing, it is important to analyze possible correlations and causes for errors and functional insufficiencies. Therefore, not only the requirements are verified here, but also more extensive tests are performed and documented. In addition, reasons for decisions made about further steps are documented here. Possible changes can be localized to all artifacts, such as changes in design/training and data curation. No best practices have been established until now on how to proceed these iterations.

Regarding the practical realization, we assume that all modifications can be traced into corresponding versions of the artifacts in a configuration management system. This configuration management defines which artifact versions belong to a particular configuration. The change request that leads to the modifications of the artifacts is documented in the change management including the corresponding artifact versions.

IV. Necessary Explicit Links

We explain the artifact links from Figure 2 and how they relate to each other.

$\boxed{1}$ *Link between System Engineering and Requirement Elicitation:* A complete and detailed ODD definition for automated driving is challenging due to an open-world context evolving over time. Despite this challenge, a definition of the environment of the system is essential. For example, it is a necessary input for a hazard analysis and risk assessment (HARA) which, in turn, is the basis for determining the safety goals. Based on the safety goals, the architecture and the ODD, the functional safety requirements are allocated to the system components and the data requirements are derived. This is usually referred to as a Top-Down Approach which is followed by most safety standards [4]. The data acquisition specification and the specification of the SDM shall be driven by the ODD as well. Without such an alignment, there is a risk of unexpected ML behavior in unknown situations.

$\boxed{2}$ *Link between Requirement Elicitation and Data Management:* Especially the adequacy of the data acquisition specification is crucial in order to achieve sufficient coverage of the semantic domain model by the data. Moreover, the labeling specification defines the required labeling quality and consequently has an impact on the performance of the component [24]. Insight from the data analysis shall be reflected in the SDM.

$\boxed{3}$ *Link between Requirements Elicitation and Design/Training:* ML requirements shall steer the Design/Training phase. If the ML model cannot satisfy certain properties on its own, a corresponding monitor may be needed, e.g., for anomaly detection. Dropout and regularization approaches during the training may be necessary due to ML robustness requirements.

$\boxed{4}$ *Link between Requirement Elicitation and Validation & Verification:* For testing, the requirements are further specified in terms of the test setup and the test oracles. As a bottom-up approach, key findings about faults, errors, and failures might be included in the requirements and SDM, if they have a significant impact on safe operation of the component. The iterations of top-down and bottom-up approaches can help to enrich requirements derived from safety goals, with DNN-specific properties. It might also happen that in the beginning of the development there are vague or hardly verifiable requirements. In this case, the evaluation shall be used to exhibit the achievable behavior which in turn can be settled iteratively as new requirements and aligned with the other components.

$\boxed{5}$ *Link between Data Management and Design & Training:* During Design & Training, we use the development dataset to create the model(s), e.g., to tune hyperparameters. Note that supervised learning approaches require labels that represent sample-wise ground truth. Missing or unbalanced development data might lead to unexpected behavior of the ML component.

$\boxed{6}$ *Link between Data Management and V & V:* We use the selected V & V data to evaluate the model using various metrics. To be able to test robustness against ODD-specific and data/sensor-specific variants, out-of-distribution data is needed, e.g., via data augmentation. Such demands may be updated during development iterations.

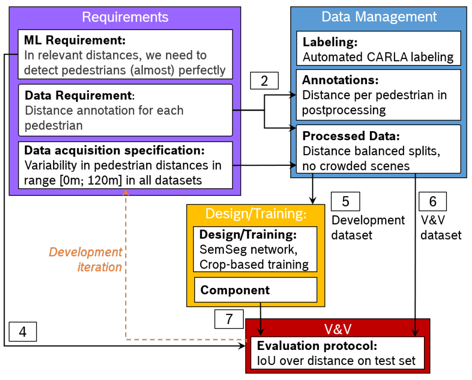
$\boxed{7}$ *Link between Design & Training and V & V:* After Design & Training, we use a resulting trained model for extensive testing. While the training optimizes the DNN to a (local) optimum of a single (possibly aggregated) loss, testing analyzes different properties of the component individually, e.g. via metrics in different subsets of the input space.

V. Concrete Application Examples with Explicit Links Between Artifacts

In this section, we discuss links between the artifacts introduced in the previous section. In particular, we leverage concrete examples from previously published works to illustrate how the links manifest themselves.

A. Example I: Evaluating Pedestrian Detection with Task-oriented Metrics

As a first example, we consider the evaluation of a pedestrian detection function for urban automated driving based on task-oriented metrics incorporating domain knowledge as presented by Lyssenko et al. [7]. The paper discusses the notion of testing leveraging a pedestrian detection function based on information about the distance of a pedestrian to the AV. The key idea is that pedestrians closer to the vehicle are more safety relevant and, thus, a misdetection becomes more critical the closer the pedestrian is. The paper demonstrates this based on an experimental setup for generating a dataset for training and test that includes pedestrians in a broad range of distances. Intuitively, the distance-based metric allows us to formulate a new ML requirement for the component



Requirements
ML Requirement:
In relevant distances, we need to detect pedestrians (almost) perfectly
Data Requirement:
Distance annotation for each pedestrian
Data acquisition specification:
Variability in pedestrian distances in range [0m; 120m] in all datasets
Data Management
Labeling:
Automated CARLA labeling
Annotations:
Distance per pedestrian in postprocessing
Processed Data:
Distance balanced splits, no crowded scenes
2
5
Development dataset
6
V&V dataset
Design/Training:
Design/Training:
SemSeg network, Crop-based training
Component
Development iteration
4
7
V&V
Evaluation protocol:
IoU over distance on test set

Fig. 3. Refinement of Artifact Links for Example 1

under test. To evaluate the ML requirement, a specific kind of V&V dataset is necessary: Additional information in the form of distance information per pedestrian is required. This necessitates new data requirements which are addressed in this particular example by the experimental setup in a simulator.

Let us concretize the relationships based on Fig. 2. A corresponding refinement of the artifacts used and their links is shown in Figure 3. Lyssenko et al. focus on automated driving functions operating in an urban environment. They use knowledge about their ODD to derive a ML requirement that states that (a) pedestrians are an important class for detection and (b) that a correct detection is more critical the closer the pedestrian is. This, in turn, leads to the data requirement of having images with pedestrians in urban road scenes in different distances in the dataset. In addition, in order to evaluate the performance over distance leads to the data requirement that all pedestrians need to be distinguishable in the ground truth labels and that their distance needs to be annotated. Lyssenko et al. use the CARLA simulator in their approach for systematically constructing datasets $\boxed{2}$. The data acquisition is performed such that the resulting data set provides a meaningful distribution of pedestrians over the range of considered distances. Here, data requirements are necessary explicitly stating, e.g., what distributions and possibly what poses, appearances, etc. should be contained in the data set. In addition, the data requirements imply the required ground truth labels. Technically, these data requirements also necessitate to postprocess the semantic segmentation ground truth and depth map provided by CARLA for identifying single pedestrians in the ground truth and to associate the distance to each of them as an instance segmentation was not available. This information is available in data management as additional annotations and should be part of a labeling specification. Thus, it is important to be able to trace the data generated for evaluating a particular ML requirement back to the ML requirement, as a change of the ML requirement may imply that the data can no longer be used.

As mentioned above, the ML requirement leads to a data requirement that, in turn, influences all datasets for training and V&V. Datasets are split randomly, however a check of distance distribution has been performed to validate a balanced distribution similar across all datasets, which is essential for this evaluation. Lyssenko et al. explicitly document the split and corresponding distributions in pedestrian distances for train/validation $\boxed{5}$ and the corresponding V&V dataset $\boxed{6}$. This is just one particular example showing that care is needed in balancing datasets w.r.t. important influencing factors [25]. As the available data set is comparably small, image augmentation based on choosing random image crops was used in training to increase diversity. The use of such techniques needs to be documented for interpreting the train and validation performance.

The aforementioned data requirement implies that (1) performance of the component under test needs to be evaluated for pedestrians with different distances and (2) that traditionally used metrics like mean average performance are not suitable for the evaluation. The first point is addressed by the path from the data requirements to the data $\boxed{2}$ and from the data to the evaluation $\boxed{6}$. As described by Lyssenko et al. [7], the second point requires specific task-oriented metrics that are tied to the ML requirements for evaluating the performance of the component under test. This is shown by $\boxed{4}$. The dashed green arrow indicates a possible development iteration. If the evaluation shows that the performance is not sufficient for certain distances, additional data for the corresponding distance range might be necessary.

This example shows that explicitly tracking links allow us to trace the effects of a change, in this case in the ML requirements. As we can see, such a change may affect various other artifacts, whether concerning the evaluation and corresponding evaluation protocol, or corresponding data requirements, which has an impact on the required V&V dataset. As a consequence, explicit documentation and suitable tracing among all artifacts is required to deal with the impact of a requirements or ODD change during development is a systematic manner.

B. Example II: Evaluating Systematic Performance Limitations of Perception Functions

As second example, we elaborate on an analysis method to assess the performance limitation of perception functions due to triggering conditions as described in [8]. The performance limitation maps (PLM) proposed in the paper are based on the concept of triggering conditions and performance limitations of the SOTIF standard [3]. They allow the identification as well as assessment of triggering conditions that lead to functional insufficiencies of the system. The impact and relation of triggering conditions (e.g. weather, reflections, etc.) on the perception metrics (false negative rate) are visualized to aid the engineers in an informed decision about the relevance for functional insufficiencies.

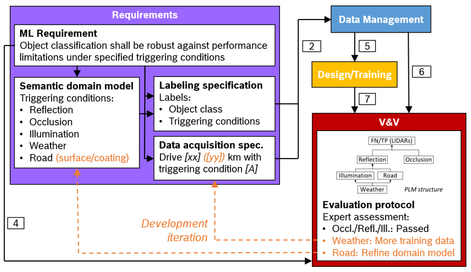


Fig. 4. Refinement of Artifact Links for Example II

The perception function discussed in [8] is a Lidar based object classification. The performance of the perception function is characterized by the false negative metric in spatial bins around the vehicle. The possible systematic factors are supplied by the perception experts in the form of a Bayesian net structure to model the casual influence of these factors on the false negative metric for each spatial grid cell. The parameters for the Bayesian net are learned from acquired data of the implemented perception function.

The method relies on collecting data for the triggering conditions and perception metrics. While the perception metrics can be directly obtained from recording system internal information, the triggering conditions have to be obtained by a labeling process. This means that these factors have to be included in the requirements by the semantic domain model (figure 4). Further, these have to be passed to the labeling specification as well as the data acquisition specification. From there the link $\boxed{2}$ to data management becomes important as the specified triggering conditions then have to be included in the acquired data to be available for analysis in the V&V process. The traceability of the link $\boxed{6}$ and $\boxed{7}$ can then be established by including a reference to the used datasets and implementation version.

The result of the PLMs is used to decide whether the implemented perception function is sufficiently robust against triggering conditions. For this the conditional PLMs are evaluated and an expert judgment is done. In the case the experts agree that sufficient robustness is achieved, no further development is necessary. More likely in early development phases the experts will identify an insufficient performance or insufficient causal structure of the Bayesian net used in the PLM due to missing or wrong triggering conditions. In that case a development iteration cycle is necessary and the importance of the links between the artifacts becomes evident

As a new triggering condition is identified or need to collect more data is identified by the experts, the links $\boxed{4}$ to the semantic domain model and data acquisition specification have to be established (dashed lines in figure 4). From the expert assessment in the evaluation protocol the link to the modified artifacts allows for traceability in future iteration steps. Going downstream through the development process again, a new dataset with these labels included in the next iteration is supplied for reassessment.

By applying these incremental development steps with traceability between the artifacts, a strong argument for a SOTIF release can be built. The systematic identification of triggering conditions of an implemented perception system with traceability between artifacts and inclusion of expert assessment demonstrates the necessary rigor to argue about sufficient reduction of the residual risk.

VI. Conclusion and summary

We provide an overview of the relevant links between the artifacts from different phases of the ML-lifecycle. We document what artifacts are relevant in the given phases and provide a detailed described how these artifacts should be connected by explicit links. We illustrate the necessity of such links based on two case studies from the domain of perception in automated driving applications. Although they involve similar artifact linkages, there are major differences in the transmission of information and in the information transmitted, since this needs to be task specific.

In example I, we emphasize the importance of the data management artifacts, which contain extracted information, data analysis, labeling, and annotations. The close linkage of the artifacts from data management and the requirement elicitation is enabled mainly between the extracted information, the data analysis and the semantic domain model and between the label annotations and the labeling specification. Based on this close linkage, the changes in the requirements are directly impacting the data. Furthermore, we show in example II the importance of a semantic domain model to document identified performance limitations and tracing them to data, requirements, and evaluation. It is crucial to establish the proposed explicit linkage starting from early development stages to ensure traceability in the lifecycle of ML based component. This supports the validity of the safety assurance case, enables adaptions of autonomous systems in an evolving open context and lays the foundation for an impact analysis.

References

[1] J. C. Knight, “Safety critical systems: challenges and directions,” in *Proceedings of the 24th International Conference on Software Engineering. ICSE 2002*, 2002, pp. 547–550.

[2] S. Burton, L. Gauerhof, and C. Heinzemann, “Making the case for safety of machine learning in highly automated driving,” in *Computer Safety, Reliability, and Security : SAFECOMP 2017 Workshops, ASSURE, DECSoS, SASSUR, TELERISE, and TIPS, Trento, Italy, September 12, 2017, Proceedings*, ser. Lecture Notes in Computer Science, S. Tonetta, E. Schoitsch, and F. Bitsch, Eds. Cham: Springer International Publishing, 2017, vol. 10489, pp. 5–16.

[3] I. Org. ISO/DIS 21448 Road vehicles Safety of the intended functionality. [Online]. Available: https://www.iso.org/standard/77490.html

[4] ISO, “ISO 26262:2018, Road vehicles - Functional safety,” 2018.

[5] L. Gauerhof, R. Hawkins, C. Picardi, C. Paterson, Y. Hagiwara, and I. Habli, “Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings,” in *Computer Safety, Reliability, and Security*, A. Casimiro, F. Ortmeier, F. Bitsch, and P. Ferreira, Eds. Springer International Publishing, 2020.

[6] C. Picardi, C. Paterson, R. Hawkins, R. Calinescu, and I. Habli, “Assurance argument patterns and processes for machine learning in safety-related systems,” in *Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI 2020)*, ser. CEUR Workshop Proceedings. CEUR Workshop Proceedings, 2020.

[7] M. Lyssenko, C. Gladisch, C. Heinzemann, M. Woehrle, and R. Triebel, “From evaluation to verification: Towards task-oriented relevance metrics for pedestrian detection in safety-critical domains,” in *Workshop on Safe Artificial Intelligence for Automated Driving*, 2021.

[8] A. Adee, R. Gansch, and Liggesmeyer, “Systematic modeling approach for environmental perception limitations in automated driving,” in *European Dependable Computing Conference*. to be published, 2021.

[9] R. Hawkins, I. Habli, T. Kelly, and J. McDermid, “Assurance cases and prescriptive software safety certification: A comparative study,” *Safety Science*, 2013.

[10] P. Bishop and R. Bloomfield, “A methodology for safety case development,” in *Industrial Perspectives of Safety-critical Systems*, F. Redmill and T. Anderson, Eds., 1998.

[11] T. Kelly and R. Weaver, “The goal structuring notation–a safety argument notation,” in *Proceedings of the dependable systems and networks 2004 workshop on assurance cases*. Citeseer, 2004, p. 6.

[12] Adelard. claims, arguments, evidence. [Online]. Available: https://claimsargumentsevidence.org/

[13] G. Schwalbe, B. Knie, T. Sämann, T. Dobberphul, L. Gauerhof, S. Raafatnia, and V. Rocco, “Structuring the safety argumentation for deep neural network based perception in automotive applications,” in *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*, A. Casimiro, F. Ortmeier, E. Schoitsch, F. Bitsch, and P. Ferreira, Eds. Cham: Springer International Publishing, 2020, pp. 383–394.

[14] L. Gauerhof, P. Munk, and S. Burton, “Structuring validation targets of a machine learning function applied to automated driving,” in *Computer Safety, Reliability, and Security*, B. Gallina, A. Skavhaug, and F. Bitsch, Eds. Springer International Publishing, 2018.

[15] R. Hawkins, C. Paterson, C. Picardi, Y. Jia, R. Calinescu, and I. Habli. Guidance on the assurance of machine learning in autonomous systems. [Online]. Available: https://www.york.ac.uk/media/assuring-autonomy/documents/AMLASv1.1.pdf

[16] U. Siddique, “Safetyops,” *arXiv preprint arXiv:2008.04461*, 2020.

[17] I. Org. ISO/AWI 34503 Road vehicles Taxonomy for operational design domain for automated driving systems. [Online]. Available: https://www.iso.org/standard/78952.html

[18] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding, supplemental material. [Online]. Available: https://www.cityscapes-dataset.com/wordpress/wp-content/papercite-data/pdf/cordts2016cityscapes-supplemental.pdf

[19] S. Albrecht, L. Gauerhof, C. Gladisch, K. Groh, C. Heinzemann, and M. Woehrle, “Testing deep learning-based visual perception for automated driving,” *Transactions on Cyber-Physical Systems*, 2021.

[20] O. Zendel, M. Murschitz, M. Humenberger, and W. Herzner, “Cv-hazop: Introducing test data validation for computer vision,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2066–2074.

[21] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth, “The reusable holdout: Preserving validity in adaptive data analysis,” *Science*, vol. 349, 2015. [Online]. Available: https://science.sciencemag.org/content/349/6248/636

[22] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[23] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuscenes: A multimodal dataset for autonomous driving,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[24] L. Gauerhof, Y. Hagiwara, C. Schorn, and M. Trapp, “Considering reliability of deep learning function to boost data suitability and anomaly detection,” in *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2020, pp. 249–254.

[25] O. Willers, S. Sudholt, S. Raafatnia, and S. Abrecht, “Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks,” in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2020, pp. 336–350.