Safety Assurance of Machine Learning for Perception Functions

Simon Burton, Christian Hellert, Fabian Hüger, Michael Mock, and Andreas Rohatschek

Abstract The latest generation of safety standards applicable to automated driving systems require both qualitative and quantitative safety acceptance criteria to be defined and argued. At the same time, the use of machine learning (ML) functions is increasingly seen as a prerequisite to achieving the necessary levels of perception performance in the complex operating environments of these functions. This inevitably leads to the question of which supporting evidence must be presented to demonstrate the safety of ML-based automated driving systems. This chapter discusses the challenge of deriving suitable acceptance criteria for the ML function and describes how such evidence can be structured in order to support a convincing safety assurance case for the system. In particular, we show how a combination of methods can be used to estimate the overall machine learning performance, as well as to evaluate and reduce the impact of ML-specific insufficiencies, both during design and operation.

S. Burton
Fraunhofer Institute for Cognitive Systems IKS, Hansastraße 32, 80686 Munich, Germany
e-mail: simon.burton@iks.fraunhofer.de

C. Hellert
Continental AG, Bessie-Coleman-Str. 7, 60549 Frankfurt am Main, Germany
e-mail: christian.hellert@continental.com

F. Hüger
Volkswagen AG, Berliner Ring 2, 38440 Wolfsburg, Germany
e-mail: fabian.hueger@volkswagen.de

M. Mock
Fraunhofer Institute for Intelligent Analysis and Information Systems IAIS, Schloss Birlinghoven 1, 53757 Sankt Augustin, Germany
e-mail: michael.mock@iais.fraunhofer.de

A. Rohatschek (✉)
Robert Bosch GmbH, Robert-Bosch-Campus 1, 71272 Renningen, Germany
e-mail: andreas-juergen.rohatschek@de.bosch.com

T. Fingscheidt et al. (eds.), *Deep Neural Networks and Data for Automated Driving*,
https://doi.org/10.1007/978-3-031-01233-4_12

1 Introduction

The objective of systems safety engineering is to avoid unreasonable risk of hazards that could lead to harm to the users of the system or its environment. The definition of an unreasonable level of risk must be determined for the specific system context in accordance with societal moral concepts and legal considerations regarding product liability. This requires a careful evaluation of the causes and impact of system failures, an evaluation of the probability of their occurrence, and the definition of strategies to eliminate or reduce the residual risk associated with such failures. Current automotive safety standards, such as ISO 26262 [ISO18] and ISO/DIS 21448 [ISO21], require a safety case to be developed for safety-related functions that forms a structured argument supported by systematically collected evidence that the appropriate level of residual risk has been achieved. The term "evidence" refers to work products created during the development, test, and operation of the system that support the claim that the system meets its safety goals.

In comparison to previous generations of vehicle systems, automated driving systems (ADS) that make use of machine learning (ML) introduce significant new challenges to the safety assurance process. Deep neural networks (DNNs), in particular, are seen as an enabling technology for ADS perception functions due to their ability to distinguish features within complex, unstructured data. This allows for the development of functions, such as pedestrian detection within crowded, urban environments that previously could not be specified and implemented based on algorithmic definitions. Paradoxically, this leads to one of the main challenges associated with the use of machine learning in safety-critical systems. The *semantic gap* [BHL+20] describes the challenge of deriving complete and consistent technical (safety) requirements on the system that fulfil potentially only implicitly understood, societal and legal expectations. The semantic gap is exacerbated in ML-based ADS due to the unpredictability and complexity of the operational domain and the reliance on properties of the training data rather than detailed specifications to derive an adequate approximation of the target function. The semantic gap can in turn lead to an unclear definition of the moral responsibility and legal liability for the system's actions as well as an incomplete assurance argument that the (potentially incompletely defined) safety goals for the system are met.

Furthermore, deep learning approaches have specific properties that limit the effectiveness of established safety measures for software. These include the inherent uncertainty in the outputs of the ML function, the opaque manner in which features are learnt by the function which is often not understandable by humans and the difficulty of extrapolating from test results due to non-linear behaviour of the function and sensitivity to small changes in the input domain. Previous work related to the safety assurance of machine learning has mainly focused on the structure of the assurance case and associated processes with respect to existing safety standards [BGH17, SQC17, GHP+20, ACP21, BKS+21]. Other work has focused on the effectiveness of specific metrics and measures on providing meaningful statements related to safety properties of the ML function [CNH+18, HSRW20, SKR+21, CKL21]. This chapter complements this work by examining in more detail the role of combining various types of evidence to form a convincing argument that quantitative acceptance criteria as required by standards such as ISO 21448 are met.

This chapter discusses the challenge of deriving suitable acceptance criteria for ML functions and describes how supporting evidence can be combined to provide a convincing safety assurance case for the system. The aggregation of evidence can be structured as an overall evidence-based safety argumentation as depicted in [MSB+21]. In the following section, we describe the challenge of deriving a set of technical safety requirements and associated acceptance criteria on ML functions. In Sect. 3, we provide a categorisation of safety evidence for machine learning and argue that an assurance case requires an appropriate combination of complementary evidence. Section 4 describes a collaborative approach between domain and safety experts for collecting and evaluating the safety evidence. Sections 5.1–5.4 provide specific examples of each category of evidence and describe the conditions under which they can provide a meaningful contribution to the assurance case. Section 6 then demonstrates how the evidence can be combined into an overall assurance case structure. The chapter concludes with a summary of open research questions and an outlook for future work.

2 Deriving Acceptance Criteria for ML Functions

This section presents approaches to risk evaluation within the safety standards and discusses how safety acceptance criteria for ML functions can be derived in line with these approaches.

2.1 Definition of Risk According to Current Safety Standards

Current standards related to the safety of vehicle systems provide little specific guidance related to the use of ML. Therefore, the approach described in this chapter will be based upon an interpretation and transfer of the principles of ISO 26262, ISO/DIS 21448, and ISO/TR 4804 to the specific task of safety assurance for machine learning-based systems. ISO 26262 defines functional safety as the absence of unreasonable risk due to hazards caused by malfunctioning behaviour of electrical/electronic systems. Malfunctioning behaviour is typically interpreted as either random hardware faults or systematic errors in the system, hardware, or software design. ISO 26262 applies a predominantly *qualitative* approach to arguing the safety of software. Safety goals for a system are derived according to a hazard and risk analysis that evaluates the risk associated with each hazard according to qualitative criteria with various categories for the risk parameters: severity, exposure, and controllability. The standard provides a method for combining these parameters to derive an overall “Automotive Safety Integrity Level” (ASIL) within a range of A to D in order of increasing risk. Functions with no safety impact are assigned the level “QM” (only standard quality management approaches are necessary). An ASIL is allocated to each safety goal derived from the hazards and to the functional and technical safety requirements derived from these until eventually software-level requirements, including associated ASILs, are determined. The standard then provides guidance on which measures to apply, depending on ASIL, to achieve a tolerable level of residual risk. By doing so, the standard avoids the need to define specific failure rates to be achieved by software but instead defines desirable properties (such as freedom from run-time errors caused by pointer mismatches, division-by-zero, etc.) and methods suited to ensuring or evaluating these properties (e.g. static analysis).

ISO/DIS 21448 addresses safety in terms of the absence of unreasonable risk due to functional insufficiencies of the system or by reasonably foreseeable misuse. In the context of ADS, this translates to a possible inability of a function to correctly comprehend the situation and operate safely, e.g. due to a lack of robustness regarding input variations or diverse environmental conditions. The risk model described by ISO/DIS 21448 can be summarised as identifying as many *known, unsafe* scenarios (where performance insufficiencies could lead to hazards) as possible so that mitigating measures can be applied, thus transforming them to *known, safe* scenarios. In addition, the number of residual *unknown, unsafe* scenarios should be minimised by increasing the level of understanding of properties of the environment that could trigger performance deficiencies. The definition of safety of the intended functionality (SOTIF), therefore, seems well suited for arguing the performance of the trained ML function over all possible scenarios within the operating domain. ISO/DIS 21448 follows a similar process to identify hazards, associated safety goals, and requirements as ISO 26262. However, instead of using risk categories according to ASILs, the standard requires the definition of *quantitative* acceptance criteria (also known as validation targets) for each hazard, which in turn can be allocated to subsystems such as perception functions. However, these acceptance criteria are not described in more detail and must, therefore, be defined according to the specific system context.

ISO/TR 4804 contains a set of guidelines for achieving the safety of automated driving systems with a focus on the Society of Automotive Engineers (SAE) levels 3–5 [SAE18]. ISO/TR 4804 defines safety acceptance criteria both in terms of a positive risk balance (the system shall be demonstrably safer than an average human driver) as well as the avoidance of unreasonable risk. In doing so, the standard recommends a combination of both qualitative and quantitative arguments. A similar philosophy is followed within this chapter when identifying and combining evidence for the safety of a machine learning function.

2.2 Definition of Safety Acceptance Criteria for the ML Function

In order to address the semantic gap described above, a systematic method of deriving a set of technical safety requirements on an ML function is needed. This should include an iterative and data-driven approach to analyse the societal and legal expectations, operating domain, and required performance on the ML function within the technical system context (see Fig. 1). In this chapter, we focus on steps 4 and 5, in particular, the systematic collection of evidence regarding the performance of an ML function to support a system-level assurance case.

In [BGH17], the authors recommend a contract-based design approach to derive specific safety requirements for the machine learning function. These requirements are expressed in the form of a set of safety guarantees that the function must fulfil and a set of assumptions which can be made in the system's environmental and technical context. This allows for a compositional approach to reasoning about the safety of the system where the guarantees of one function can be used to justify the assumptions on the inputs of another. This requires a suitable definition of safety at the level of abstraction of the ML function. For example, a superficially defined requirement, such as "detect a pedestrian on the road", would need to be refined to define which characteristics constitute a pedestrian and from which distance and level of occlusion pedestrians should be detected. This process of requirements elicitation should include the consideration of a number of stakeholder perspectives and sources of data. This could include current accident statistics and the consideration of ethical guidelines for AI as proposed by the European Commission [Ind19]. It is to be expected that for any particular automated driving function, a number of such contracts would be derived to define different properties of the ML function related to various safety goals, where each contract may also be associated with different quantitative acceptance criteria and sets of scenarios in the operating domain.

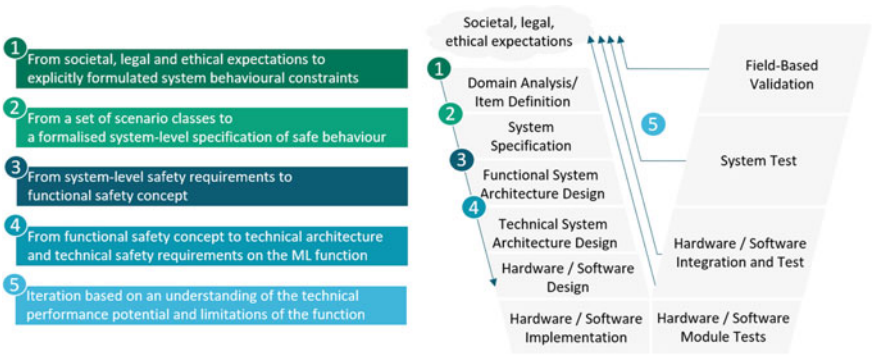

1
From societal, legal and ethical expectations to explicitly formulated system behavioural constraints
2
From a set of scenario classes to a formalised system-level specification of safe behaviour
3
From system-level safety requirements to functional safety concept
4
From functional safety concept to technical architecture and technical safety requirements on the ML function
5
Iteration based on an understanding of the technical performance potential and limitations of the function
Societal, legal, ethical expectations
1
Domain Analysis/ Item Definition
2
System Specification
3
Functional System Architecture Design
4
Technical System Architecture Design
Hardware / Software Design
Hardware / Software Implementation
Field-Based Validation
5
System Test
Hardware / Software Integration and Test
Hardware / Software Module Tests

Fig. 1 Iterative steps during the systems engineering process to bridge the semantic gap associated with the definition of technical safety requirements on ML functions

Table 1 Example safety contract for a pedestrian detection

Input domain	The set of all possible situations within an urban environment in which pedestrians could be within the range of the vehicle's sensors
Assumptions	Safety-relevant pedestrians have a size 50 cm $\leq size \leq$ 250 cm and are within a range of 3 m $\leq range \leq$ 50 m
Guarantees	Bounding box accuracy per frame in the sequence shall be within 20 cm of the ground truth, and the pedestrian must be detected whenever the occlusion through other objects is $<$50%. The confidence score for successful classification per frame shall be $\geq$70%

Table 1 contains an example of a (not necessarily complete) safety contract for a bounding-box pedestrian detection model. This contract is defined within the scope of a system architecture where the quality of the camera signal acting as input to the function is either known or can be determined during development and where a sensor fusion and monitoring component receives the result of the ML-based detection algorithm and performs time-series plausibility checks on the results and compares with other sensor modalities such as RaDAR and LiDAR.

A *qualitative* approach to defining the assurance targets for the ML function could involve determining a suitable combination of development and test methods to be applied that are assumed to lead to a correct implementation of the function. However, due to typical failure modes and performance limitations of ML, an absolute level of correctness in the function is infeasible. Instead, *quantitative* assurance targets, in line with the requirements of ISO/DIS 21448, are required that would define an acceptable limit to the *probability* that guarantees cannot be met. For example, an acceptance criterion for pedestrian detection based on the remaining accuracy rate (RAR) metric [HSRW20] could be formulated as *"An RAR of 95% is achieved and residual errors are distributed equally across all scenarios"*, leading to the probability of a single pedestrian being undetected by both the ML function and the sensor fusion/monitor component being sufficiently low. The remaining accuracy rate is defined in [HSRW20] as the proportion of inputs where a confidence score above a certain threshold (e.g. 70%) leads to a true positive detection. Thus, the definition of the parameters for the quantitative acceptance targets must be defined based on a detailed understanding of the performance limits of both the ML function and the capabilities of the sensor fusion/monitoring component.

3 Understanding the Contribution of Safety Evidence

3.1 A Causal Model of Machine Learning Insufficiencies

Ideally, to demonstrate that the ML function meets its performance requirements, the probability of failure will be directly measured, for example, based on a sufficiently

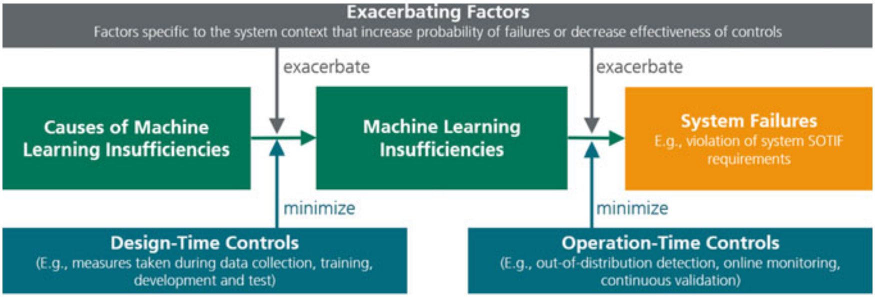

Exacerbating Factors
Factors specific to the system context that increase probability of failures or decrease effectiveness of controls
exacerbate
exacerbate
Causes of Machine Learning Insufficiencies
Machine Learning Insufficiencies
System Failures
E.g., violation of system SOTIF requirements
minimize
minimize
Design-Time Controls
(E.g., measures taken during data collection, training, development and test)
Operation-Time Controls
(E.g., out-of-distribution detection, online monitoring, continuous validation)

Fig. 2 Causal model of ML-related SOTIF risks demonstrating the relationships between system failures, machine learning insufficiencies, their causes, and exacerbating factors as well as the role of control measures to minimise the probability of hazardous system failures

large number of representative tests. However, this approach is feasible only for trivial examples. Therefore, multiple sources of evidence are required to reason about the presence of performance limitations and the ability of the system to detect and control such failures during operation.

The dependability model of Avizienis et al. [ALRL04] forms the foundation of many safety analysis techniques. In this model, the risk of a system violating its objectives is determined by analysing the propagation path from causes of individual *faults* in the system that could lead to an *erroneous* system state which in turn leads to a *failure* to maintain the system's objectives. Risk can thus be controlled by either eliminating the causes of faults or by preventing them from leading to potentially hazardous erroneous states of the system.

Figure 2 summarises how this approach to causal analysis can be applied to the problem of determining the safety-related performance of an ML function and is inspired by the safer complex systems framework presented in [BMGW21]. A system failure can be defined as a condition where the safety contract as defined in Table 1 is not met. An erroneous state leading to such a failure would relate to the presence of insufficiencies in the machine learning function which, in turn, could be caused by either faults in its execution or limitations in the training data. Examples for the latter are scalable oversight [AOS+16], uncertainty in the input domain, such as distributional shift [AOS+16], or the inherent inability of the ML approach to accurately represent the target function. Measures to improve the safety of the function can be categorised into those applied during design time to reduce the probability of insufficiencies in the trained model and those applied during operation in order to reduce the impact of residual insufficiencies. Both types of controls may be undermined by exacerbating factors specific to the system context, for example, the difficulty in collecting balanced and sufficiently complete training data due to the scarcity of critical scenarios or the difficulty of developing effective monitoring approaches due to the need for safe-operational fallback concepts. This model of causality for ML-related safety of the intended functionality (SOTIF) risks is now used to determine the following contributions of safety evidence that will be illustrated in the next subsection.

3.2 Categories of Safety Evidence

Based on the causal model of SOTIF-related risk introduced above, the following four categories of evidence can now be defined. This category of evidence corresponds to development work products and results of verification and validation activities that support the argument of an acceptably low level of residual risk associated with ML insufficiencies:

(1) Confirmation of residual failure rates: This category of evidence provides a direct estimate of the performance of the machine-learning component, e.g. in terms of false negative rates, or the intersection-over-union (IoU) related to bounding box accuracy. However, due to non-linear behaviour in the function and the limited number of samples that can be realistically tested, such evidence is unlikely to provide a statistically relevant estimation. This category of evidence must, therefore, be used in combination with other measures that increase confidence in the ability to extrapolate the results to the entire input space that fulfils the contract's assumptions.

(2) Evaluation of insufficiencies: This category of evidence is used to directly indicate the presence or absence of specific ML insufficiencies that could not only lead to a violation of the safety contract, but could also undermine confidence in the category 1 evidence. The properties that would be evaluated by this category of evidence would include prediction uncertainty, generalisation, brittleness, fairness, and explainability.

(3) Evaluation of the effectiveness of design-time controls: This category of evidence is used to argue the effectiveness of design-time measures to increase the performance of the function or to reduce the presence of insufficiencies. In many cases, a direct correlation between the design-time measures and the performance of the function may not be measured, leading to qualitative arguments for the effectiveness of the measures. However, metrics can be used to measure the level of rigour by which the measures have been applied. These could include properties of training data and test data, or measures to increase the explainability of the trained model.

(4) Evaluation of the effectiveness of operation-time controls: This category of evidence is used to demonstrate the effectiveness of operation-time measures to either increase the performance of the function or to reduce the impact of insufficiencies. Examples of such could be a measurement of the effectiveness of out-of-distribution detection to identify inputs that are outside of the training distribution (and beyond the assumptions of the safety contract) such that they can be discarded.

In the following sections, we illustrate each category with specific examples and evaluate the conditions under which the evidence can provide a significant contribution to the assurance case.

4 Evidence Workstreams—Empowering Experts from Safety Engineering and ML to Produce Measures and Evidence

In order to identify, develop, and evaluate effective evidence related to the various categories defined above, we propose to follow the procedure of so-called evidence *workstreams*: experts from machine learning, safety, testing, and data working together according to the procedure depicted in Fig. 3. The main purpose of the process is to help demonstrate the effectiveness of the methods, metrics, and measures to be applied more generally. Related processes are also found in current literature. Picardi et al. [PPH+20] propose an engineering process to generate evidence at each stage of the ML life cycle. Apart from the process in general, different patterns are described that can be instantiated during the ML life cycle. Furthermore, a three-layer process model is described by McDermid and Jia [MJ21], featuring a collaborative model to bring together the expertise from the ML and safety domain to generate evidence for the assurance case. In this work, there are some case studies, but detailed steps for the process are missing.

The evidence workstreams proposed in this chapter enable a joint cooperation of different competencies and bring together new innovative methods and structured approaches rather than having separate work on each topic. An important key to the successful creation of an assurance case is the continuous interaction of the different contributors. Before explaining the procedure, we, therefore, define four roles for the contributors:

- The *method developer* is responsible to implement measures that mitigate specific insufficiencies of an ML component. In addition, this role specifies data

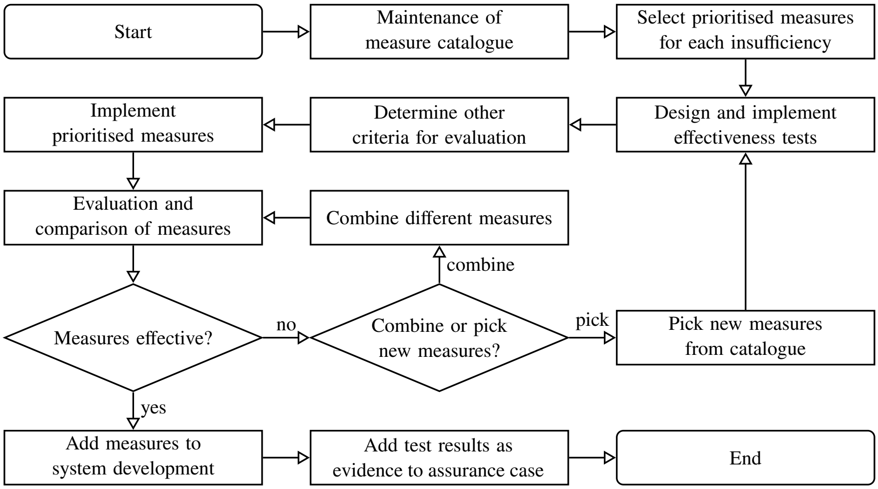

Start
Maintenance of measure catalogue
Select prioritised measures for each insufficiency
Design and implement effectiveness tests
Determine other criteria for evaluation
Implement prioritised measures
Evaluation and comparison of measures
Combine different measures
combine
Measures effective?
no
Combine or pick new measures?
pick
Pick new measures from catalogue
yes
Add measures to system development
Add test results as evidence to assurance case
End

Fig. 3 Flowchart of the evidence workstream process

requirements for the implemented measures and defines metrics for evaluating the effectiveness of the measure.

- Apart from the method developer specifying data requirements, a *data engineer* is responsible for creating and managing the dataset throughout the development process as well as performing data analyses to estimate the data coverage.
- In order to test the effectiveness of the measures, the *test engineer* develops test specifications including required test data and metrics. These tests should then be performed by the method developer to demonstrate the effectiveness of the method.
- Finally, the *safety engineer* ensures that the developed measures and respective test results are valid and generate the evidence for the assurance case.

The process starts by creating and maintaining a catalogue of potential design-time and run-time measures to mitigate identified insufficiencies, which can be known a priori (see Sect. 5.2) or identified during the development phase. Here, *method developers* and *safety engineers* are involved in the maintenance and also perform a prioritisation of measures for each insufficiency. After the selection of measures, the *test engineer* designs and implements effectiveness tests. Furthermore, safety-aware metrics for the test are defined by the *method developer* and *safety engineer*, in addition to other criteria for the evaluation. Importantly, the *data engineer* needs to evaluate the database in order to allow for a statistical significance analysis for the designed tests. Afterwards, the prioritised measures are implemented following an evaluation and comparison according to the specified tests. Once the measures show sufficient effectiveness, they can be added to the system development and the *safety engineer* can use the test results as evidence for the assurance case. If a measure was not effective, then we propose the two following options:

- Firstly, measures can be combined and optimised to increase the effectiveness. After combination, a re-evaluation is performed.
- Secondly, new measures can be picked from the catalogue and the test design and implementation step is repeated.

Additionally, an iteration of the safety requirement derivation and system architecture design process could lead to recommendations for additional components or adjusted specifications in order to mitigate remaining insufficiencies.

5 Examples for Evidence

This section discusses examples for the different categories of safety evidence introduced in Sect. 3.2, which can be obtained by the process introduced in Sect. 4.

5.1 Evidence from Confirmation of Residual Failure Rates

The quantification of model performance or residual failure rates of a machine learning (ML) component is a crucial step in any ML-based system, especially for safety-critical applications. Typically, the performance of ML components is measured by metrics that calculate a mean performance over a specific test dataset. In [PNdS20] and [PPD+21], commonly used metrics for object detection are compared, including mean average precision (mAP) and mean intersection-over-union (mIoU). However, for safety-critical systems, only evaluating the mean performance is not sufficient. On the one hand, there are unknown risks introduced by residual failure rates, on the other hand, failures are not weighted by their relevance. To counteract this, for example, in [HSRW20], safety-aware metrics are proposed in the literature to incorporate uncertainty to evaluate the remaining accuracy rate and remaining error rate. In [CKL21], safety-aware metrics for semantic segmentation are proposed, putting emphasis on perturbation and relevance. Volk et al. [VGvBB20] propose a safety metric that incorporates object velocity, orientation, distance, size, and a damage potential. Furthermore, ISO/TR 4804 [ISO20] proposes a generic concept, whereby safety-aware metrics need to be evaluated for perception components realised with deep neural networks. In addition, it should be noted that the validity of metrics depends strongly on the utilised dataset. In general, the creation of safety-aware metrics is still an open field in the domain of automated driving and for perception components in particular. Based on the references and insights above, the following considerations should be taken into account to use performance metrics as evidence:

- Mean performance metrics should be evaluated within a specified input space.
- Statistical significance of the metrics value should be evaluated and shown by measuring dataset coverage.
- The failure rate related to specific classes of errors should be evaluated, including an analysis of their causes.

The evidence for performance or accuracy evaluation could be in the form of a test report, where the specification of the input space and the data coverage evaluation is defined. Apart from detection or classification accuracy, localisation precision is also an important factor and is often dependent on the accuracy. In addition to a summary of the safety-aware metrics, an analysis of various classes of errors, their potential causes and impact within the system should be included in the test report, allowing for a systematic evaluation of the residual risk associated with the function at a system level.

The following example provides an intuition of conditions that have to be met so that a performance metric can be used as evidence convincingly. The basis of nearly every performance metric is the calculation of the confusion matrix, containing the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) rates. Here, we consider the case of a 2D bounding box pedestrian detection on images for automated driving. In this case, the number of TNs is not reasonable since there will be a nearly infinite amount of those bounding boxes. In order to compute the confusion matrix, the intersection-over-union (IoU) metric is typically used to determine if a detection can be considered as a TP, FP, or FN by using a threshold. This threshold, on the one hand, defines the localisation error per object which must not be exceeded for the detection to be counted as TP, and, on the other hand, also influences the mean accuracy on a test dataset. Therefore, in the test report, there should be an argument about why a certain threshold has been chosen. Furthermore, typically FP and FN are weighted equally, but for a safety analysis, FN might be more important and not all FN might be relevant. Hence, the test report should define relevant FN or FP based on the system architecture and assigned component requirements by introducing specific weights. Of course, performance metrics alone cannot give enough evidence to argue the safety of a perception system for automated driving. In addition, the performance needs to be evaluated in case of rarely occurring situations, and also the robustness against perturbations needs to be considered.

5.2 Evidence from Evaluation of Insufficiencies

An essential feature in the development of deep neural networks during training lies in the purely data-driven parameter fitting process without expert intervention: The deviation of the output (for a given parameterisation) of a neural network from a ground truth is measured. The loss function is chosen in such a way that the parameters depend on it in a differentiable way. As part of the gradient descent algorithm, the parameters of the network are adjusted in each training step depending on the derivative of the deviation (backpropagation). These training steps are repeated until some stopping criterion is satisfied. In this procedure, the model parameters are determined without an expert assessment or semantically motivated modelling. This has significant consequences for the properties of the neural network:

- Deep neural networks (DNNs) are largely opaque for humans and their calculations cannot be interpreted. This represents a massive limitation for systematic testing or formal verification.
- Deep neural networks are susceptible to harmful interference: Perturbations could be manually induced changes in the data (adversarial examples) or real-word corruptions (e.g. sensor noise, weather influences, certain colours, or contrasts by sensor degeneration).
- It is unclear to which input characteristics an algorithm sensitises. The execution of neural networks in another domain (training in summer, execution in winter, etc.) sometimes reduces the functional quality dramatically.

This leads to DNN-specific insufficiencies and safety concerns as described in detail by Willers et al. [WSRA20], Sämann et al. [SSH20], and Schwalbe et al. [SKS+20]. In our evidence strategy, we utilise metrics to evaluate the insufficiencies and use specific values and justified bounds as specific evidence. In the following, this proce-

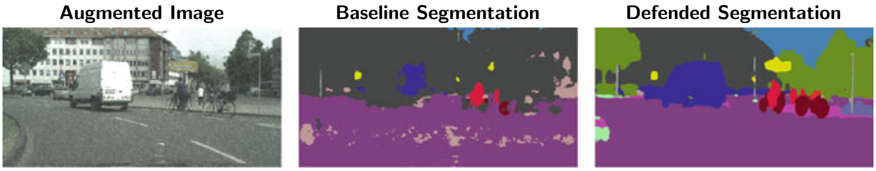


Fig. 4 Visualisation of the robustness of a semantic segmentation model: Rain augmentation via Hendryck's augmentations [HD19] on Cityscapes [COR+16] (left), baseline performance on the corrupted image (middle) and performance of a robustified model via AugMix with Jensen–Shannon divergence (JSD) loss [HMC+20] (right)

dure is shortly explained for the example of brittleness of DNNs. The strategies for the other insufficiencies follow basically the same pattern, but the elaboration would go beyond the scope of this chapter.

Measuring robustness of deep neural networks (DNNs): Regarding brittleness, the specific evidence that we want to define are test results achieving the required performance even under reasonable perturbations. If the required performance is achieved for all required conditions, we call the DNN "robust". To evaluate the robustness with respect to real-world corruptions, we suggest to use both, recorded and tagged real-word effects as well as augmented perturbations such as noise, blur, colour shift, etc., via Hendryck's augmentations [HD19] as depicted in Fig. 4. The required parameters for the test conditions are extracted from operational design domain (ODD), sensor, and data analysis.

5.3 *Evidence from Design-Time Controls*

Design-time controls systematically complement and integrate evidence gained via overall performance considerations or regarding specific insufficiencies. The major goal is to integrate the various aspects and metrics that have to be considered into a holistic view, allowing the AI developer, firstly to incorporate measures that are overall effective, such as architectural choices, and parameter optimisations, or data coverage measures, and secondly, to handle potential trade-offs between different optimisation targets and requirements. It is important to note that the design-time controls are to be applied in an iterative workflow during development, aiming to provide sufficient evidence for the overall safety argumentation. As a specific example of a design-time control, we sketch how developers and safety engineers can be supported by a visual analytics tool during the design and development phase.

Understanding DNN predictions via visual analytics: An important contribution to a complete safety argumentation can be made via methods that support humans in understanding and analysing why an AI system is coming to a specific decision. Insights into the inner operations of a DNN can increase trust into the entire application of that neural network. However, in a safety argumentation, it is not sufficient to

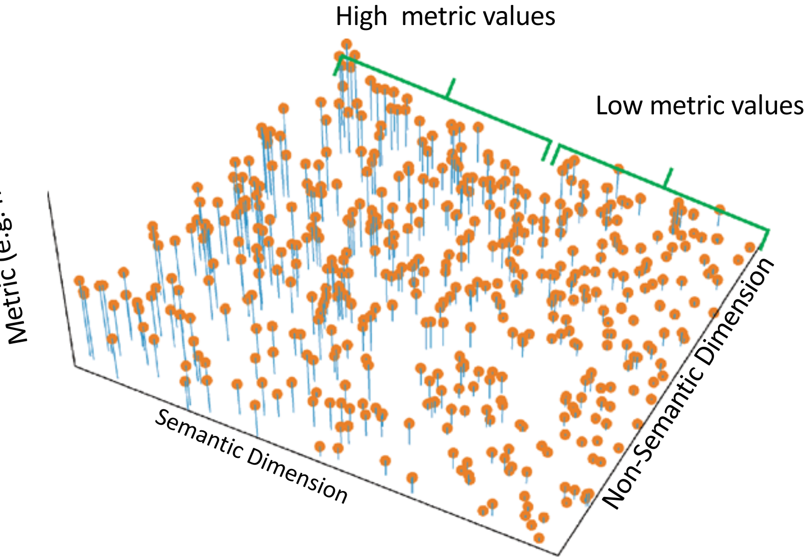


Fig. 5 Landscape of DNN performance over semantic and non-semantic dimensions

"explain" single decisions of a network, for example, in some kind of "post analysis" in case of an accident, which might have been caused by a failure of a DNN-based pedestrian detection, it should be possible to provide arguments about the understanding of the inner operations of a neural network considering a huge set of input test data at design time. The obvious solution for achieving this would be completely replacing the DNN with an interpretable model. However, this often is not feasible considering the trade-off with overall performance of the network. Ideally, such a "design-time understanding" of the overall behaviour of a DNN would result in an understanding of the network performance over a complete landscape spanned by semantic and non-semantic dimensions of the test data as illustrated in Fig. 5.

Figure 5 shows an idealised view of the performance of the DNN in a selected relevant metric, see explanations in Sect. 5.1, per input test data point over selected semantic and non-semantic dimensions. In the case of pedestrian detection in the automotive context, semantic dimensions may refer to the semantics of a scene description in the operational design domain, such as pedestrian attributes or environmental conditions, while non-semantic dimensions may refer to technical image effects, such as blurring or low contrast, which are supposed to have an influence on the performance of the DNN. Such a view can support the human to identify systematic weaknesses of a DNN in terms of human-understandable dimensions. To be able to create a view as depicted in Fig. 5, a human user, either in the role of a DNN developer, safety engineer, or auditor, is faced with two challenges:

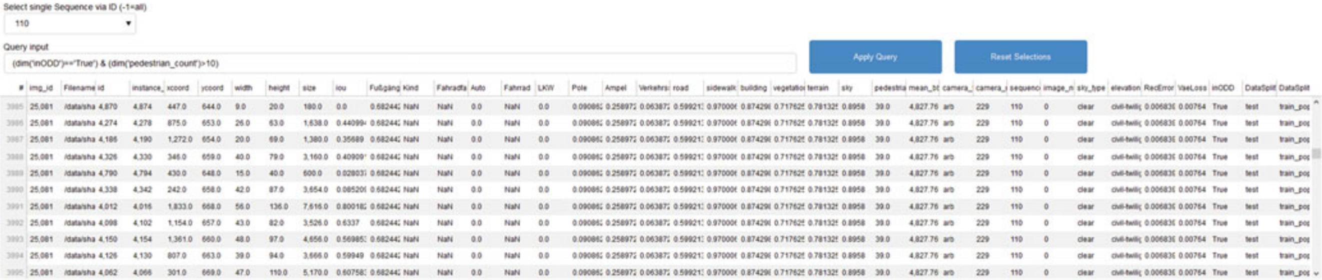
- The sheer amount of test data needed to reach significant insights usually exceeds the cognitive capacity of the human brain.

- Finding and selecting the relevant dimensions that actually influence the network performance among all possible dimensions is far from being trivial, especially taking into account that the performance of a DNN mostly depends on a combination of different dimensions and usually cannot be explained by considering one dimension alone.

In order to overcome these two challenges, tool support is needed. In [SMHA21], an approach is described to support the human user in finding and evaluating machine learning models by means of visual analytics. In this approach, tool support is provided to enable the user to perform visual analyses over huge amounts of data in such a way that a quick and intuitive understanding of the nature of the underlying data and network performance can be gained. Such an initial understanding usually enables the user to create hypotheses about the relevance of semantic and non-semantic dimensions, which then can be checked or refined interactively. Human understanding about semantics helps in a semantic understanding of the DNN in an iterative process.

Figures 6 and 7 show a snapshot of the visual analytics tool being used to investigate the performance of a DNN for pedestrian detection based on a semantic segmentation of images. The so-called metadata that indicates the values regarding a specific dimension of an input image (such as number of pedestrians, brightness of the image, and overall detection performance in the image) and the values regarding specific pedestrians in the image (such as size, position in the image, detection performance for this pedestrian) is listed in a tabular interface as illustrated in Fig. 6. The user can issue arbitrary queries against this table to select interesting subsets of the data. The plots shown in Fig. 7 give an additional overview of various attributes. Selected images, predictions, or ground truth can be visualised as shown in Fig. 9.

Figure 8 shows an exemplary scatterplot of the pixel size of a pedestrian in an image versus the detection performance measured in IoU (intersection-over-union) for this pedestrian. The goal is to find out whether certain aspects, such as pixel size of the pedestrian in the image, have an effect on the detection performance of the underlying DNN. It can be seen that there is a general tendency for pedestrians with larger height (more to the left) to achieve higher (better) detection performance. However, there exist some large pedestrians with relatively low detection performance (as highlighted by the selection box in Fig. 8). The visual analytics tool allows for interactively selecting such sets of interesting points from a plot for further analysis just by drawing a selection box around the points of interest. Selected images will then be visualised in the drill down view of the tool shown in Fig. 9. In this particular example, some of the selected images were showing “pedestrians” riding bicycles as depicted in Fig. 9b). The human analyst could now use the visual analytics tool to further investigate the hypothesis that cyclists are not detected well enough by the DNN.



Select single Sequence via ID (-1=all)

110 ▾

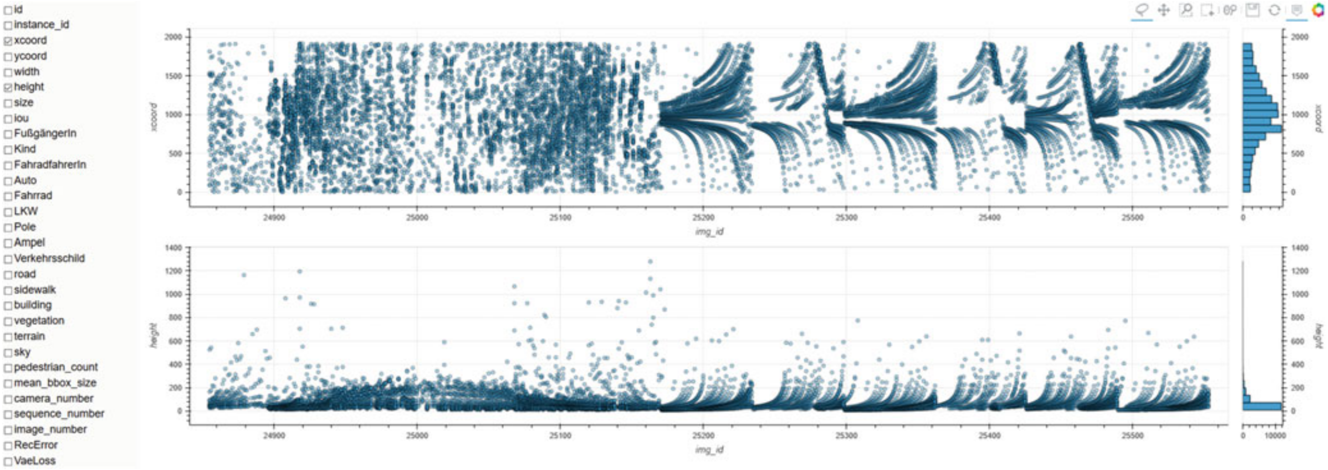
Query input

(dim('inODD')=='True') & (dim('pedestrian_count')>10)

#	img_id	Filename	id	instance_	xcoord	ycoord	width	height	size	iou	Fußgäng	Kind	Fahradfa	Auto	Fahrrad	LKW
3985	25,081	/data/sha	4,870	4,874	447.0	644.0	9.0	20.0	180.0	0.0	0.68244	NaN	NaN	0.0	NaN	0.0
3986	25,081	/data/sha	4,274	4,278	875.0	653.0	26.0	63.0	1,638.0	0.44099	0.68244	NaN	NaN	0.0	NaN	0.0
3987	25,081	/data/sha	4,186	4,190	1,272.0	654.0	20.0	69.0	1,380.0	0.35689	0.68244	NaN	NaN	0.0	NaN	0.0
3988	25,081	/data/sha	4,326	4,330	346.0	659.0	40.0	79.0	3,160.0	0.40909	0.68244	NaN	NaN	0.0	NaN	0.0
3989	25,081	/data/sha	4,790	4,794	430.0	648.0	15.0	40.0	600.0	0.02803	0.68244	NaN	NaN	0.0	NaN	0.0
3990	25,081	/data/sha	4,338	4,342	242.0	658.0	42.0	87.0	3,654.0	0.08520	0.68244	NaN	NaN	0.0	NaN	0.0
3991	25,081	/data/sha	4,012	4,016	1,833.0	668.0	56.0	136.0	7,616.0	0.80018	0.68244	NaN	NaN	0.0	NaN	0.0
3992	25,081	/data/sha	4,098	4,102	1,154.0	657.0	43.0	82.0	3,526.0	0.6337	0.68244	NaN	NaN	0.0	NaN	0.0
3993	25,081	/data/sha	4,150	4,154	1,361.0	660.0	48.0	97.0	4,656.0	0.56985	0.68244	NaN	NaN	0.0	NaN	0.0
3994	25,081	/data/sha	4,126	4,130	807.0	663.0	39.0	94.0	3,666.0	0.59949	0.68244	NaN	NaN	0.0	NaN	0.0
3995	25,081	/data/sha	4,062	4,066	301.0	669.0	47.0	110.0	5,170.0	0.60758	0.68244	NaN	NaN	0.0	NaN	0.0

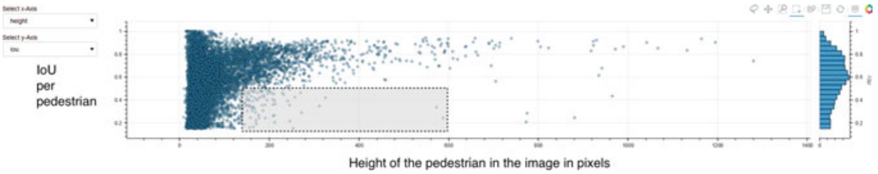
#	Pole	Ampel	Verkehrs	road	sidewalk	building	vegetatio	terrain	sky	pedestria	mean_bt	camera_	camera_i	sequenc	image_n	sky_type	elevation	RecError	VaeLoss	inODD	DataSplit	DataSplit
3985	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3986	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3987	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3988	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3989	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3990	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3991	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3992	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3993	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3994	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po
3995	0.09086	0.25897	0.06387	0.59921	0.97000	0.87429	0.71762	0.78132	0.8958	39.0	4,827.76	arb	229	110	0	clear	civil-twili	0.00683	0.00764	True	test	train_po

Fig. 6 Visual analytic tool for analysing DNN performance and weaknesses: tabular query interface on meta data



□ id
□ instance_id
☑ xcoord
□ ycoord
□ width
☑ height
□ size
□ iou
□ FußgängerIn
□ Kind
□ FahradfahrerIn
□ Auto
□ Fahrrad
□ LKW
□ Pole
□ Ampel
□ Verkehrsschild
□ road
□ sidewalk
□ building
□ vegetation
□ terrain
□ sky
□ pedestrian_count
□ mean_bbox_size
□ camera_number
□ sequence_number
□ image_number
□ RecError
□ VaeLoss

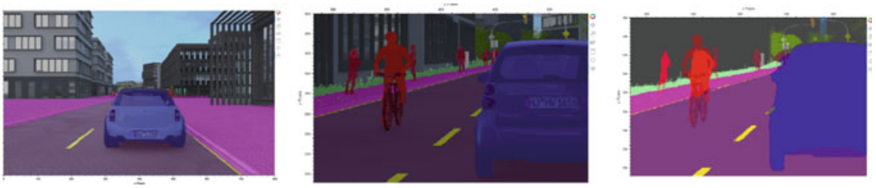
Fig. 7 Visual analytic tool for analysing DNN performance and weaknesses: plots of selected attributes



Select x-Axis
height

Select y-Axis
iou

Fig. 8 Plot of the pedestrian detection performance (IoU) in dependence of the height of the pedestrian in the image in pixels



(a) Overlay of input image and ground truth

(b) Overlay of ground truth (left) and predictions (right): selected critical images often contain cyclists

Fig. 9 Visual analytic tool for analysing DNN performance and weaknesses: drill down view showing **a** a sample image, and **b** critical images selected from the correlation plot shown in Fig. 8 (Images: BIT Technology Solutions)

5.4 Evidence from Operation-Time Controls

Operation-time controls are typically used to mitigate unacceptable risk occurring during run-time of the system. For example, when the performance of a perception component for automated driving degenerates, controls could become active to mitigate the degeneration by handing vehicle controls back to a human driver or by using alternative perception components. The operation-time controls are particularly important for automated driving, since such systems operate in the real world. As during design time, not all possibly occurring situations can be foreseen, architectural solutions are required.

In this section, we give two examples of how evidence from operation-time controls can be derived. The first example deals with out-of-distribution detection and the second one goes into detail for uncertainty estimation.

Out-of-distribution detection: One of the major causes of ML insufficiencies is the fact that the data distribution of an operational domain is unknown. Therefore, it is only possible to sample from this unknown distribution and approximate this distribution by statistical approaches or machine learning (ML) techniques. During the construction of the safety contract, a certain input domain is defined at a semantic level, which directly leads to a gap to the real data distribution of the operational domain. One of the resulting ML insufficiencies is that there is an unknown behaviour when the ML function is presented with samples that are not within the training data distribution. Therefore, a requirement could be that leaving the operational design domain (ODD) has to be detected at run-time. Such a requirement could introduce a monitoring component using complementary methods. Three example methods could be:

- When the ODD defines certain areas of operation, e.g. urban intersections, localisation information in combination with map data can be utilised to detect leaving the ODD.
- Some sensors can directly measure an environmental state, such as a rain sensor, which can be used to set bounds according to the ODD to detect whether the ODD is left.
- Out-of-distribution methods [HG17, XYA20] can be used to detect a distributional shift of the input samples during run-time with respect to the database used for training a ML model for perception or planning.

For the last method, there could furthermore be two types of evidence. Firstly, the effectiveness of the methods should be shown by a test report including an evaluation of a metric that indicates the separation precision between in- and out-of-distribution samples. Secondly, it has to be shown that the training and test data used to train the out-of-distribution detector is sufficient.

Uncertainty estimation: Uncertainty estimation methods can also be applied as operation-time controls. Uncertainty is an inherent property of any machine learning model, which may result from either aleatoric sources, such as a non-deterministic behaviour of the real world or issues in the data or labelling, or epistemic sources, referring to the inherent limitations of the machine learning model itself. Uncertainty estimation methods aim to enable the DNN to indicate the uncertainty related to a specific DNN prediction given a specific input at run-time. This may also contribute to out-of-distribution detection under the assumption that the uncertainty increases when the DNN is applied outside of its training data distribution. This assumption of course must be rigorously tested in order to provide the corresponding evidence. Among popular uncertainty estimation methods, methods based on Monte-Carlo (MC) dropout [GG16] receive particular attention in embedded applications as they approximate the performance of Bayesian Networks and hence usually outperform techniques purely based on post-processing calibration, and come with an acceptable run-time overhead (compared to, e.g. full Bayesian networks or deep ensembles). Although originally intended for capturing epistemic uncertainty only, it can be extended to capture aleatoric uncertainty as well [KG17, SAP+21].

6 Combining Safety Evidence in the Assurance Case

While the insights sketched above surely help to understand weaknesses and improve the development of the DNN under investigation, the step towards arguing evidence in an assurance case still has to be performed. According to the principles of ISO 26262, ISO/DIS 21448, and ISO/TR 4804, the assurance case shall state in a convincing

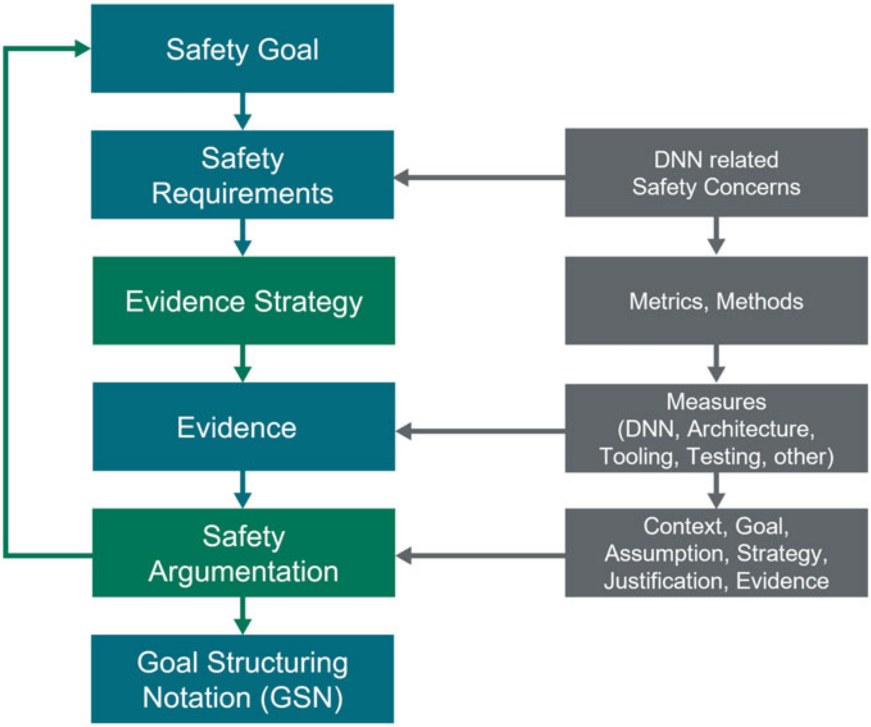

Safety Goal
Safety Requirements
Evidence Strategy
Evidence
Safety Argumentation
Goal Structuring Notation (GSN)
DNN related Safety Concerns
Metrics, Methods
Measures (DNN, Architecture, Tooling, Testing, other)
Context, Goal, Assumption, Strategy, Justification, Evidence

Fig. 10 Assurance case development: Integration of evidence into an assurance case

way: *"The system is safe because..."*. However, the assurance of DNNs leads to several problems, since this technology requires new paradigms in development. The software is no longer explicitly developed. Instead, the neural network is trained and the network's behaviour is implicitly influenced by the training models and data. The combination of safety evidence in the assurance case provides central elements for a holistic assurance strategy.

The core aspect of safety argumentation is to show that the mitigation of insufficiencies was successful. If the insufficiency is reduced to an acceptable level, this provides evidence to be used in the safety argumentation. As shown in Fig. 10, one possibility to combine safety evidence in the assurance case is to start on the top level with the definition of safety goals. These are goals that define mandatory steps to avoid hazards. Then the safety requirements are refined step-by-step based on the described causal model of SOTIF-related risk using the categories of evidence. This is supported by considering DNN-related safety concerns. Moreover, several metrics are defined to show the effectiveness of measures that mitigate the effects of insufficiencies. The goal structuring notation (GSN) can be used to assemble evidence collected from various methods as they were presented in Sects. 5.1, 5.2, 5.3, and 5.4 to provide a structured overall safety argumentation. The GSN visualises the elements of the safety argumentation. An assurance case can be presented in a clear and structured way in the GSN. A GSN tree consists of three central elements: the argumentation goal, the description of the argumentation strategy, and the evidence. These three elements are supported by assumptions, justifications, and context information. A central aspect is the iterative nature of this technique to refine understanding of insufficiencies in the function. Further iterations are started on the top level (definition of safety goals).

7 Conclusions

Machine learning is an enabler technology for automated driving, especially for, but not limited to, the required perception components. However, assuring safety for such components is a challenge as standard safety-argumentation concepts are not sufficient to capture the inherent complexity and data-dependency of functions based on machine learning. For example, a component realised with an ML function is formally hard to describe, which in turn makes it especially difficult to define appropriate evidence for the safety argumentation. Therefore, we introduce different types of evidence as a structuring guidance: evidence from confirmation of residual failure rates, evidence from evaluation of insufficiencies, evidence from design-time controls, and evidence from operation-time controls.

In order to create appropriate evidence, a process is required to ensure that knowledge from different domains (machine learning, safety, and testing) are brought together. Therefore, we propose the process of evidence workstreams to define evidence in a structured way. Furthermore, we show how to integrate evidence into an assurance case.

One of the main questions still remains open: Can a convincing assurance case be constructed? We argue "yes" but only by explicitly acknowledging the insufficiencies in the ML function within the system design, and by being able to determine the residual failures with sufficient confidence. This implies directly defining an acceptable residual risk that is acknowledged by social acceptance and legal conditions, which is an open challenge.

Future research topics might be how to combine multiple quantitative and qualitative pieces of evidence into the safety argumentation w.r.t. a given system architecture and how to balance them. Moreover, there is demand for derivation of evidence from the appropriate coverage of all possible situations within the operational design domain (ODD) based on a structured ground context including tests. Further, the iterative and dynamic process of constructing the assurance case (or "continuous assurance") requires work on formal models of the assurance case and on the continuous evaluation of the assurance case.

Acknowledgements The research leading to these results is funded by the German Federal Ministry for Economic Affairs and Energy within the project "Methoden und Maßnahmen zur Absicherung von KI-basierten Wahrnehmungsfunktionen für das automatisierte Fahren (KI Absicherung)". The authors would like to thank the consortium for the successful cooperation.

References

[ACP21] R. Ashmore, R. Calinescu, C. Paterson, Assuring the machine learning lifecycle: desiderata, methods, and challenges. ACM Comput. Surv. (CSUR) **54**(5), 1–39 (2021)

[ALRL04] A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr, Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. Dependable Secur. Comput. (TDSC) **1**(1), 11–33 (2004)

[AOS+16] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, D. Mané, Concrete problems in AI safety (2016), pp. 1–29. arXiv:1606.06565

[BGH17] S. Burton, L. Gauerhof, C. Heinzemann, Making the case for safety of machine learning in highly automated driving, in *Proceedings of the International Conference on Computer Safety, Reliability, and Security (SAFECOMP)* (Trento, Italy, 2017), pp. 5–16

[BHL+20] S. Burton, I. Habli, T. Lawton, J. McDermid, P. Morgan, Z. Porter, Mind the gaps: assuring the safety of autonomous systems from an engineering, ethical, and legal perspective. Artif. Intell. **279**, 103201 (2020)

[BKS+21] S. Burton, I. Kurzidem, A. Schwaiger, P. Schleiss, M. Unterreiner, T. Graeber, P. Becker, Safety assurance of machine learning for chassis control functions, in *Proceedings of the International Conference on Computer Safety, Reliability, and Security (SAFECOMP)* (York, UK, 2021), pp. 149–162

[BMGW21] S. Burton, J.A. McDermid, P. Garnett, R. Weaver, safety, complexity, and automated driving: holistic perspectives on safety assurance. Computer **54**(8), 22–32 (2021)

[CKL21] C.-H. Cheng, A. Knoll, H.-C. Liao, Safety metrics for semantic segmentation in autonomous driving (2021), pp. 1–8, arXiv:2105.10142

[CNH+18] C.-H. Cheng, G. Nührenberg, C.-H. Huang, H. Ruess, H. Yasuoka, Towards dependability metrics for neural networks, in *Proceedings of the ACM/IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)* (Beijing, China, 2018), pp. 43–46

[COR+16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The Cityscapes dataset for semantic urban scene understanding, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Las Vegas, NV, USA, 2016), pp. 3213–3223

[GG16] Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: representing model uncertainty in deep learning, in *Proceedings of the International Conference on Machine Learning (ICML)* (New York, NY, USA, 2016), pp. 1050–1059

[GHP+20] L. Gauerhof, R.D. Hawkins, C. Picardi, C. Paterson, Y. Hagiwara, I. Habli, Assuring the safety of machine learning for pedestrian detection at crossings, in *Proceedings of the International Conference on Computer Safety, Reliability, and Security (SAFECOMP)* (Virtual conference, 2020), pp. 197–212

[HD19] D. Hendrycks, T. Dietterich, Benchmarking neural network robustness to common corruptions and perturbations, in *Proceedings of the International Conference on Learning Representations (ICLR)* (New Orleans, LA, USA, 2019), pp. 1–15

[HG17] D. Hendrycks, K. Gimpel, A baseline for detecting misclassified and out-of-distribution examples in neural networks, in *Proceedings of the International Conference on Learning Representations (ICLR)* (Toulon, France, 2017), pp. 1–12

[HMC+20] D. Hendrycks, N. Mu, E.D. Cubuk, B. Zoph, J. Gilmer, B. Lakshminarayanan, Augmix: a simple data processing method to improve robustness and uncertainty, in *Proceedings of the International Conference on Learning Representations (ICLR)* (Virtual Conference, 2020), pp. 1–15

[HSRW20] M. Henne, A. Schwaiger, K. Roscher, G. Weiss, Benchmarking uncertainty estimation methods for deep learning with safety-related metrics, in *Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI)* (New York, NY, USA, 2020), pp. 1–8

[Ind19] Independent High-Level Expert Group on Artificial Intelligence. *Ethics Guidelines for Trustworthy AI* (European Commission, 2019)

[ISO18] ISO. *ISO 26262: Road Vehicles—Functional Safety* (International Organization for Standardization (ISO), 2018)

[ISO20] ISO. *ISO/TR 4804: Road Vehicles—Safety and Cybersecurity for Automated Driving Systems—Design, Verification and Validation* (International Organization for Standardization (ISO), 2020)

[ISO21] ISO. *ISO/DIS 21448: Road Vehicles—Safety of the Intended Functionality* (International Organization for Standardization (ISO), 2021)

[KG17] A. Kendall, Y. Gal, What uncertainties do we need in Bayesian deep learning for computer vision? in *Proceedings of the Conference on Neural Information Processing Systems (NIPS/NeurIPS)* (Long Beach, CA, USA, 2017), pp. 5574–5584

[MJ21] J. McDermid, Y. Jia, Safety of artificial intelligence: a collaborative model, in *Proceedings of the Workshop on Artificial Intelligence Safety* (Virtual Conference, 2021), pp. 1–8

[MSB+21] M. Mock, S. Scholz, F. Blank, F. Hüger, A. Rohatschek, L. Schwarz, T. Stauner, An integrated approach to a safety argumentation for AI-based perception functions in automated driving, in *Proceedings of the International Conference on Computer Safety, Reliability, and Security (SAFECOMP) Workshops* (York, UK, 2021), pp. 265–271

[PNdS20] R. Padilla, S.L. Netto, E.A.B. da Silva, A survey on performance metrics for object-detection algorithms, in *Proceedings of the IEEE International Conference on Systems, Signals and Image Processing (IWSSIP)* (Niteiro, Brazil, 2020), pp. 237–242

[PPD+21] R. Padilla, W.L. Passos, T.L.B. Dias, S.L. Netto, E.A.B. da Silva, A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics **10**(3), 1–28 (2021)

[PPH+20] C. Picardi, C. Paterson, R.D. Hawkins, R. Calinescu, I. Habli, Assurance Argument Patterns and Processes for Machine Learning in Safety-Related Systems. In: *Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI)* (New York, NY, USA, 2020), pp. 1–8

[SAE18] SAE International. *SAE J3016: Surface Vehicle Recommended Practice—Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles* (SAE International, 2018)

[SAP+21] J. Sicking, M. Akila, M. Pintz, T. Wirtz, A. Fischer, S. Wrobel, A novel regression loss for non-parametric uncertainty optimization, in *Proceedings of the Symposium on Advances in Approximate Bayesian Inference* (Virtual conference, 2021), pp. 1–27

[SKR+21] F. Schwaiger, M.H. Fabian Küppers, F.S. Roza, K. Roscher, A. Haselhoff, From black-box to white-box: examining confidence calibration under different conditions, in *Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI)* (Virtual Conference, 2021), pp. 1–8

[SKS+20] G. Schwalbe, B. Knie, T. Sämann, T. Dobberphul, L. Gauerhof, S. Raafatnia, V. Rocco, Structuring the safety argumentation for deep neural network based perception in automotive applications, in *Proceedings of the International Conference on Computer Safety, Reliability, and Security (SAFECOMP) Workshops* (Virtual Conference, 2020), pp. 383–394

[SMHA21] E. Schulz, M. Mock, S. Houben, M. Akila, *ScrutinAI: an iterative workflow for the semantic analysis of DNN predictions* (Technical report, Fraunhofer IAIS, St. Augustin, 2021)

[SQC17] R. Salay, R. Queiroz, K. Czarnecki, An analysis of ISO 26262: using machine learning safely in automotive software (2017), pp. 1–6. arXiv:1709.02435

[SSH20] T. Sämann, P. Schlicht, F. Hüger, Strategy to increase the safety of a DNN-based perception for HAD systems (2020), pp. 1–14. arXiv:2002.08935

[VGvBB20] G. Volk, J. Gamerdinger, A. von Betnuth, O. Bringmann, A comprehensive safety metric to evaluate perception in autonomous systems, in *Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC)* (Virtual Conference, 2020), pp. 1–8

[WSRA20] O. Willers, S. Sudholt, S. Raafatnia, S. Abrecht, Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks, in *Proceedings of the International Conference on Computer Safety, Reliability, and Security (SAFECOMP) Workshops* (2020), pp. 336–350

[XYA20] Z. Xiao, Q. Yan, Y. Amit, Likelihood regret: an out-of-distribution detection score for variational auto-encoder (2020), pp. 1–19. arXiv:2003.02977