IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 October 2024, accepted 16 November 2024, date of publication 25 November 2024, date of current version 9 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3506333

STANDARDS

A Systematic Approach to Enhancing ISO 26262 With Machine Learning-Specific Life Cycle Phases and Testing Methods

PADMA IYENGHAR[1,2], EMIL GRACIC[3], AND GREGOR PAWELKE[3]
[1]innotec GmbH, 70794 Filderstadt, Germany
[2]Faculty of Engineering and Computer Science, University of Applied Sciences, 49076 Osnabrück, Germany
[3]CARIAD SE, 10587 Berlin, Germany

Corresponding author: Padma Iyenghar (padma.iyenghar@innotecsafety.com)

ABSTRACT This paper presents a systematic approach to enhancing ISO 26262, a widely adopted standard for automotive functional safety, by integrating Machine Learning (ML)-specific life cycle phases and testing methods for Automotive Safety Integrity Level (ASIL) A/B. With the increasing incorporation of ML techniques in automotive systems, the current ISO 26262 framework reveals significant gaps in addressing ML-specific safety requirements. While ISO/DPAS 8800 provides an approach for developing AI systems that meet some safety properties, it does not provide a mapping concept for ASIL classification of ML systems. Furthermore, given the complexity of ML techniques in automotive systems, issues such as interpretability—critical for transparency and accountability—along with robustness and uncertainty handling, pose significant challenges that are not fully addressed by ISO 26262 and ISO/DPAS 8800. This study identifies and addresses these gaps by defining three additional life cycle phases: prepare data, train ML model, and deploy ML model. For each life cycle phase, we establish desired properties such as robustness, uncertainty handling, and interpretability, and propose suitable methods to achieve these properties. We adopt a rigorous evaluation framework inspired by IEC 61508 to assess the effectiveness of these methods. Since the method recommendations of ISO 26262 for ML-based products are incomplete, the approach presented in this paper provides critical guidance and room for expert assessment and independent certification, ensuring solid and reliable recommendations. This systematic, clear, uniform development procedure not only supports product teams in achieving their safety goals but also facilitates the certification process, reducing ambiguity and enhancing the overall safety and reliability of ML-based automotive systems.

INDEX TERMS Artificial intelligence (AI), data model, automotive functional safety, certification process, embedded systems, evaluation framework, interpretability, ISO 26262, life cycle phases, machine learning (ML), method recommendations, model deployment, robustness, safety-critical systems, software life cycle, systematic approach, testing methods, V-model.

I. INTRODUCTION

The rapid advancements in automotive technology have led to an increased integration of Machine Learning (ML) techniques into safety-critical systems. ISO 26262 [1], the international standard for functional safety of electrical and electronic systems in production automobiles, has been the key in guiding the development of safe automotive systems. It addresses various aspects of vehicle development, from item definition to system-level, hardware, and software implementation, as well as various supporting and management processes. However, when it comes to using ML as a software development technology, the standard does not provide specific guidelines for ML-based systems. As ML becomes state-of-the-art, its integration into functional safety contexts requires new design and testing concepts

The associate editor coordinating the review of this manuscript and approving it for publication was Liang-Bi Chen.

that differ significantly from traditional code-centric or model-based approaches. In contrast, the ISO/DPAS 8800 [2] draft proposal, currently under development, defines a general safety framework for Artificial Intelligence (AI) and ML in the automotive industry but lacks specific guidelines for various Automotive Safety Integrity Levels (ASIL). For a broader understanding of functional safety beyond software development, ISO/IEC TR 5469 [3], which addresses AI-functional safety and ML systems, offers a comprehensive view, including management activities, tool usage, and hardware integration. However, it does not provide ML-specific life cycle phases, testing methods, or mapping to various ASILs. Thus, despite the recent developments, that attempt to provide frameworks for AI and ML in safety contexts, they do not fully address the gaps left by ISO 26262, particularly in defining ML-specific life cycle phases and testing methods. This highlights the need for a comprehensive extension of ISO 26262 to accommodate the unique challenges posed by ML. This paper aims to address these gaps by proposing a systematic approach to enhance ISO 26262, integrating ML-specific life cycle phases and testing methods, from a software perspective.

Nevertheless, inputs from higher levels of development are also crucial for understanding their impact on ML safety and the potential derivation of the respective ASIL. Two important considerations include:

- Considering input from Hazard Analysis and Risk Assessment (HARA) regarding sensor faults and limitations, as these directly implicate insufficiencies in the ML software components.
- Considering input from the Software Safety Analysis conducted at the software architecture level to understand the relationship between ML components and other traditional software components. This relationship can mean that the ML component may benefit from monitoring or fault detection by other software, or conversely, that the ML component might need to handle faults generated by other software parts.

Other aspects of functional safety, such as management activities, overall requirements management, and hardware integration, are equally critical but are out of the scope of this study. ISO/IEC TR 5469 [3] can be referenced for a more comprehensive understanding of these areas in the broader context of AI-based systems.

A. IDENTIFIED GAPS

As stated above, this paper proposes a systematic approach to extend ISO 26262 by integrating ML-specific life cycle phases and testing methods from a software perspective. Through a systematic literature review, discussed in section II, several critical gaps have been identified in integrating ML into the ISO 26262 framework. The gaps w.r.t. ML development relate (but are not exclusively restricted) to:

- Software requirements
- Data Aspects
- Training Processes
- Model Deployment

Details regarding the identified gaps can be found in section II. These gaps highlight the necessity for an updated approach that incorporates ML-specific considerations to ensure the safety and reliability of modern automotive systems.

B. CONTRIBUTION

This paper leverages existing standards IEC 61508 [4] and ISO 26262 [1] to establish a robust framework for incorporating ML-specific life cycle phases and testing methods. The approach includes identifying gaps in these standards, defining new life cycle phases, and mapping and evaluating test methods. This paper introduces several key innovations, to address the aforementioned identified gaps in ISO 26262 for ML-based automotive systems, which are listed below:

1) **Extended life cycle phases**: We propose three additional life cycle phases specific to ML:
   - Prepare data
   - Train ML model
   - Deploy ML model

   These phases are seamlessly integrated with the existing ISO 26262 framework, ensuring comprehensive coverage of ML-specific requirements.
2) **Desired Properties and Methods**: For each newly defined life cycle phase, we establish critical desired properties, such as robustness, uncertainty handling, and interpretability. Additionally, we propose suitable methods to achieve these properties. In this paper, we ensure that each desired property is covered by at least one test method across the three life cycle phases. While additional test methods may be explored in future work, the priority in this study was to guarantee comprehensive coverage of the key properties, aligning with our testing methods-centric focus.
3) **Systematic and Rigorous Evaluation Framework**:
   - We adopt a systematic and rigorous evaluation framework inspired by IEC 61508 (Part 3, Annex C) to assess the effectiveness of the proposed methods.
   - This framework assigns rigor levels (R1 and R2) to different methods, providing a structured approach that ensures ML models meet the required desired safety properties, which in the end support the overall product safety.
   - This approach facilitates expert assessment and independent certification, paving the way for these methods to be integrated into future versions of ISO 26262.
4) **Mapping and Evaluation of Methods**:
   - Adapt methods from existing standards.
   - Evaluate which desired properties are supported by each method.

- Evaluate the effectiveness of each method (e.g. by assigning rigor levels R1 and R2) to fulfill the assigned desired properties, thereby creating mapping tables to align desired properties with test methods.

5) **Outputs and Recommendations**:

- Defined inputs, methods, and objectives for each life cycle phase, similar to the structure used in ISO 26262.
- Provide method recommendations for ASIL A/B based on rigorous evaluation.

By addressing these gaps, our contributions provide a clear, systematic approach to incorporating ML-specific requirements into ISO 26262. This not only supports product teams in achieving their safety goals but also streamlines the certification process, offering reliable and expert-backed recommendations. Ultimately, this work contributes to systematically considering the characteristics of ML-based automotive systems in the development, fostering greater confidence not only in their functionality but also in achieving the overall goal for software development w.r.t ISO 26262, i.e. reducing the probability of systematic errors.

Please note that since the method recommendations of ISO 26262 for the development of ML-based products are incomplete, there is some room for interpretation regarding the suitability of methods for the ML-specific life phases. In our opinion, if the introduced ML life cycle phases and the novel ML testing concept are assessed and accepted by an independent certification body, a solid and reliable recommendation is created. A clear recommendation reduces the room for interpretation and can establish a uniform testing concept that supports product teams in achieving their safety goals.

Additionally, this paper introduces a novel systematic ML testing concept, which enables the release of ML products that conform to ASIL A and ASIL B (a gap that is not addressed in [1], [2] and [5]). Please note that ASIL C and ASIL D are out of scope for this paper. A brief background on ASIL levels is available in Section II-A.

Furthermore, this study focuses on the most widely used and applied ML technologies in automotive systems, specifically Neural Networks (NN). This includes Convolutional Neural Networks (CNN), Feed-Forward Neural Networks (FFNN), and Recurrent Neural Networks (RNN). It considers both supervised learning and transfer learning methodologies. Thus, the usage of the term *ML* throughout the paper reflects the study's specific focus on ML technologies used in automotive systems, particularly the ones mentioned above. Broader AI concepts and other ML methods, such as decision trees or Bayesian networks, as well as reinforcement and unsupervised learning approaches, are out of the scope of this study. However, future work may extend the framework to cover these additional ML categories.

C. ORGANIZATION

Following this introduction, the remainder of this paper is organized as follows. Section II deals with background and related work. Section III outlines our proposed systematic approach. Sections IV, V and VI deal with the evaluation and results for ML data model, trained model and deployed model respectively. Section VII provides a discussion and conclusion.

II. BACKGROUND AND RELATED WORK

In this section, a brief background is provided on ISO 26262 and the current gaps in ISO 26262 pertaining to ML development are elaborated. Following this, an overview of the existing standards relevant to ML is provided, along with a mention of the gaps that are pertinent to the scope of this paper. Related work is presented next, with a literature review and summary of the gaps identified. This section ends with a list of definitions for some of the relevant terms used in this paper.

A. ISO 26262

ISO 26262 [1] is an international standard for functional safety in the automotive industry. It provides a comprehensive framework for the development and production of safety-critical systems, ensuring that they meet stringent safety requirements throughout their life cycle. Part 6 of ISO 26262 focuses specifically on the software development process, outlining a V-model that defines development activities, verification, and validation steps.

The V-model in Part 6 (cf. Fig. 1) guides the development from software requirements specification to design, implementation, integration, testing, and release. This structured approach helps in identifying and mitigating potential safety risks at each stage of the software life cycle. However, this model primarily addresses traditional software systems and does not accommodate the complexities and unique requirements of ML-based systems. As automotive systems increasingly incorporate ML, the limitations of ISO 26262 in addressing these advanced technologies become more apparent.

ISO 26262 introduces the concept of ASIL to classify safety requirements based on the severity, exposure, and controllability of potential hazards. These levels range from ASIL A (the lowest) to ASIL D (the highest), and each level helps determine the necessary rigor in development and validation.

- ASIL A corresponds to systems with minimal safety impact, where failure might lead to minor disruptions.
- ASIL B indicates a moderate safety risk, necessitating more stringent safety measures than ASIL A.
- ASIL C and ASIL D apply to systems where failure could result in severe or life-threatening consequences, requiring the highest levels of verification and validation.

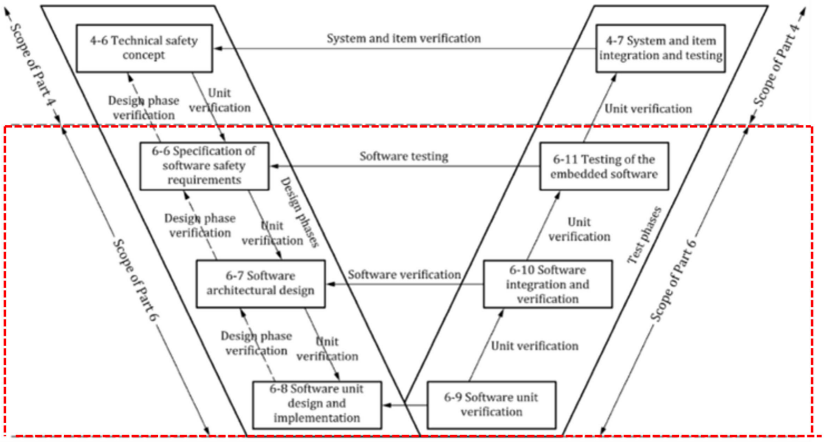

FIGURE 1. Life cycle phases of ISO 26262 [1].

As the ASIL level increases, so does the rigor in the design, verification, and validation processes, ensuring that more critical systems are developed with more robust safety mechanisms. Please note that, in ISO 26262, the concept of rigor levels such as R1 and R2 are not formally introduced or explicitly defined. However, the standard inherently implements a rigor-based approach through ASIL levels (ASIL A-D), which dictate the level of effort, thoroughness, and safety measures required for different system components based on the potential hazard they present. It should be noted that the concept of different rigor levels has been introduced by IEC 61508. Further details on this can be found in section II-C5.

B. CURRENT GAPS IN ISO 26262 PERTAINING TO ML DEVELOPMENT

Despite its comprehensive nature, a detailed analysis of the ISO 26262 standard reveals several notable gaps regarding ML-based systems, including:

1) **Software Requirements:** The standard does not adequately cover ML-specific safety properties such as robustness, uncertainty handling, and interpretability. These properties are critical for ensuring that ML models operate safely and reliably under diverse and unpredictable conditions.
2) **Data Aspects:** Effective ML functionality relies heavily on the quality and management of data. ISO 26262 lacks guidelines on handling various data aspects, including data collection, preprocessing, labeling, and validation. These steps are crucial for training ML models that are both accurate and reliable.
3) **Training Processes:** ML training processes involve stochastic methods and iterative optimization, often using frameworks like TensorFlow or PyTorch. The existing ISO 26262 framework does not provide guidance on how to manage and document these processes to ensure that the resulting models meet safety standards. Emerging technologies such as Large Language Models (LLMs) and Large Multimodal Models (LMMs) represent significant advancements in the ML field and are increasingly influencing ML practices. These foundational models introduce new complexities in terms of scale, interpretability, and data dependencies. While our current focus is on traditional ML processes, future work will need to address the implications of these cutting-edge models, particularly regarding their integration with safety standards such as ISO 26262.
4) **Model Deployment:** Transforming trained ML models from high-level programming languages like Python into embedded C/C++ code for deployment in automotive systems is another area not covered by the current standard. This transformation is critical for ensuring that the deployed models maintain their integrity and safety properties in a real-world operational environment.

C. OVERVIEW OF EXISTING STANDARDS RELEVANT TO ML

Several other standards and guidelines provide valuable insights into developing and testing ML models, although they are not specifically tailored to automotive applications. These include:

1) ISO/IEC 29119-11

ISO/IEC 29119-11 [6] is a part of the ISO/IEC 29119 software testing standard series, which focuses on the testing of AI systems. This standard provides comprehensive guidelines for testing ML-based systems, covering various aspects of verification and validation of ML models. It emphasizes the importance of testing for properties such as robustness, interpretability, and bias, which are crucial for ensuring the reliability and safety of AI systems.

In the context of automotive applications, these properties are particularly important due to the safety-critical nature of automotive systems. Robustness ensures that the ML models can handle unexpected inputs and operate reliably under diverse conditions. Interpretability is vital for understanding

the decision-making process of the ML models, which is essential for debugging and validating the system's behavior. The ISO/IEC 29119-11 standard also addresses the need for systematic testing approaches that can be adapted to the specific requirements of ML models used in automotive systems. By integrating the principles and methods from ISO/IEC 29119-11 into the ISO 26262 framework, we can enhance the testing and validation processes for ML-based automotive systems.

2) ISO/IEC TR 5469

ISO/IEC TR 5469 [3] provides a comprehensive framework for addressing functional safety in AI systems. It focuses on the properties, risk factors, and methods associated with AI systems, particularly in safety-critical functions. This standard explores three key areas: the use of AI to realize safety-related functions, the use of non-AI safety functions to ensure the safety of AI-controlled equipment, and the application of AI systems in the design and development of safety-related functions.

In the context of automotive applications, ISO/IEC TR 5469 is particularly relevant as it provides methods for integrating AI into safety-critical systems while ensuring functional safety. The standard emphasizes managing risks, transparency, validation, and safety assurance, which are crucial for AI-driven automotive systems where failures could have severe consequences. Integrating these principles into the ISO 26262 framework enhances the safety and reliability of ML-based automotive systems by addressing gaps in risk assessment, validation, and testing, and ensuring robust, transparent, and safe AI deployment in automotive environments.

3) ISO/IEC 5338

ISO/IEC 5338 [7] defines a comprehensive set of life cycle processes for AI systems, building upon existing standards like ISO/IEC/IEEE 15288 [8] and ISO/IEC/IEEE 12207 [9], while adding AI-specific processes derived from ISO/IEC 22989 [10] and ISO/IEC 23053 [11]. This standard provides guidelines for managing AI systems throughout their life cycle, addressing key aspects such as system definition, control, execution, and improvement.

In the context of automotive safety, ISO/IEC 5338 is particularly relevant as it offers a structured approach for integrating AI-specific life cycle processes into safety-critical systems. These processes align with the gaps identified in ISO 26262, such as model training, data handling, and deployment. By incorporating the systematic management of AI systems into safety frameworks like ISO 26262, we can better address challenges related to robustness, transparency, and traceability of ML models in automotive applications. Leveraging the principles and methods from ISO/IEC 5338 will enhance the development, testing, and validation of AI-based systems, supporting our proposal for extending ISO 26262 to include ML-specific life cycle phases. Thus, the gaps pertaining to the definition of life cycle phases are intended to be closed by introducing three new life cycle phases inspired by [12] and compatible with those defined in ISO/IEC 5338 [7].

4) ISO/DPAS 8800

This is a Draft Publicly Available Specification (DPAS) that provides principles and methods for the development of reliable AI systems. It emphasizes key aspects such as transparency, explainability, and robustness, offering guidelines that can be adapted to ensure the safety and reliability of ML models in various applications, including automotive systems.

Transparency and explainability are critical in the automotive context, where understanding the behavior and decision-making process of ML models is essential for validating their performance and ensuring safety. Robustness, on the other hand, ensures that the ML models can maintain their performance and reliability across a wide range of operating conditions and scenarios.

ISO DPAS 8800 outlines a systematic approach for developing AI systems that meet these critical properties. This approach includes methodologies for data management, model development, and testing, which can be tailored to the specific requirements of automotive applications. By incorporating these methodologies into the ISO 26262 framework, we can address the unique challenges posed by ML models in automotive systems, ensuring that they meet the stringent safety standards required for ASIL A/B conformance.

Standards like ISO 26262 and ISO PAS 8800 address functional safety in AI systems but lack focus on validating ML models in real-world conditions. ASAM OpenSCENARIO [13], while not AI-specific, offers a framework for simulating driving scenarios to test ML models in autonomous vehicles. Scenario-based testing complements the proposed ML life cycle phases by enabling real-time validation of model robustness and uncertainty in safety-critical environments.

ISO 15288 [8] forms the basis for many key systems engineering processes, offering a comprehensive framework for managing the life cycle of systems. Recent developments, such as [14], emphasize the potential of leveraging model-based approaches to formalize processes and standards. This concept of modeling can serve as an extension of our work, where life cycle phases for ML systems can be formalized and integrated into established systems engineering frameworks like ISO 26262. Such an approach would provide enhanced traceability, rigor, and consistency, thereby ensuring the effective application of ML in safety-critical automotive systems.

5) IEC 61508

Known as the "mother" of functional safety standards, IEC 61508 provides a comprehensive framework for ensuring the functional safety of electrical, electronic, and programmable

electronic systems. It is a crucial standard that serves as the basis for numerous sector-specific functional safety standards, such as those used in the automotive, process, and machinery industries. The standard is structured into seven parts, each addressing different aspects of functional safety. Part 1 to Part 3 cover the general requirements, requirements for electrical/electronic/programmable electronic (E/E/PE) systems, and software requirements, respectively. Parts 4 to 7 provide supporting information, including definitions, the use of safety integrity levels (SILs), guidelines on the application of IEC 61508 for specific industries, and examples of methods and techniques.

Part 3, Annex C of IEC 61508 is particularly relevant for the work discussed in this paper as it offers a detailed methodology for evaluating the effectiveness of safety measures. This annex outlines the safety life cycle, detailing the activities and tasks necessary to achieve functional safety. It provides a systematic approach to assessing safety functions, assigning safety integrity levels, and ensuring that all safety-related systems and components meet the required safety standards. The safety life cycle includes phases such as concept, overall scope definition, hazard and risk analysis, overall safety requirements, safety-related system design and development, and operation and maintenance.

The evaluation framework provided in Annex C can be adapted and tailored to assess ML-specific methods within the ISO 26262 framework. This involves a systematic approach to defining and validating ML-specific life cycle phases and testing methods. By assigning rigor levels (R1 and R2) to different methods, the framework ensures that the methods are evaluated for their effectiveness in supporting the achievement of desired safety properties. Rigor level 1 (R1) indicates a lower level of rigor, suitable for less critical functions, while rigor level 2 (R2) indicates a higher level of rigor, suitable for more critical functions. This rigorous evaluation process helps in establishing clear and reliable recommendations, reducing the room for interpretation and ensuring that ML-based systems meet the stringent safety requirements necessary for ASIL A/B conformance.

Thus, by leveraging the methodology outlined in IEC 61508, particularly Part 3, Annex C, we can develop a more robust and comprehensive approach to integrating ML-specific life cycle phases and testing methods into ISO 26262. This integration enhances the applicability of ISO 26262 to modern automotive systems, ensuring that ML-based safety-critical systems are developed and assessed with the highest levels of rigor and effectiveness.

6) SUMMARY OF GAPS IDENTIFIED

In addition to the gaps in ISO 26262 pertaining to ML development summarized in section II-B, a thorough analysis of related standards such as ISO/IEC 29119-11, ISO/IEC TR 5469, ISO/IEC 5338, and ISO DPAS 8800 provides valuable insights into ML development but reveals several critical gaps concerning the integration of ML into automotive safety systems. While these standards offer guidance on general ML development, they do not address the unique safety-critical requirements of automotive applications, particularly in terms of life cycle phases, testing, and deployment.

First, these standards do not adequately cover ML-specific software requirements, including key safety properties such as robustness, uncertainty handling, and interpretability. Additionally, there is insufficient guidance on dataset validation and inadequate coverage of the stochastic nature of ML training processes. Furthermore, the transformation of ML models from high-level programming languages like Python to embedded C/C++ code or specific hardware configurations remains under-addressed. In terms of validation and testing, although standards like ISO/IEC 29119-11 provide testing methodologies for AI systems, they fall short of addressing the unique challenges of testing ML models in safety-critical automotive contexts. Lastly, while standards like ISO/IEC 5338 provide general life cycle processes for AI systems, they do not fully account for the specific life cycle phases required for ML development in automotive applications. Thus, an analysis of these ML-related, non-automotive-specific standards offers key insights into the broader challenges of ML integration. Leveraging these insights enables the development of a more comprehensive approach to incorporating ML-specific life cycle phases and testing methods into ISO 26262.

D. RELATED WORK

This section discusses related work in industry projects and literature pertaining specifically to integrating ML technologies into the ISO 26262 framework. We carried out a Systematic Literature Review (SLR) focusing on works that cover the critical aspects of ML integration. Please note that these are the main aspects pertain to the main topic of our paper, namely, *a systematic approach to enhancing ISO 26262 with ML-specific life cycle phases and testing methods*. The SLR followed the steps outlined below:

1) **Search Strategy:** A comprehensive search strategy was defined to retrieve relevant studies from academic databases and industry reports. The search terms included combinations of keywords, such as: (a) ''ISO 26262'' AND ''ML safety'', (b) ''AI/ML automotive standards'' AND ''functional safety'' and (c) ''machine learning life cycle'' AND ''automotive safety''. The databases searched include IEEE Xplore, ACM Digital Library and Google Scholar, as well as relevant industry white papers.
2) **Inclusion and Exclusion Criteria:** The literature review focused on studies that specifically address the application of ML technologies within automotive safety-critical systems. Research that delved into standards such as ISO 26262, PAS 8800, and SOTIF, particularly in the context of integrating ML, was prioritized. Additionally, papers proposing extensions or adaptations to these functional safety standards to accommodate AI/ML-specific requirements were included. Studies that discussed general-purpose

AI/ML without a clear focus on the automotive domain were excluded from the review. Furthermore, papers that lacked empirical validation or did not address safety-related aspects in their discussions were also omitted to ensure relevance to the scope of enhancing automotive safety standards.

3) **Study Selection and Screening:** After conducting the initial search and the application of the inclusion/exclusion criteria, 19 studies were shortlisted for further analysis. The selected studies were primarily from the past 5 years, ensuring the review was focused on recent developments.
4) **Data Extraction and Synthesis:** The data extraction process involved capturing key details from the selected studies, such as, (a) Identified gaps in the existing safety frameworks for ML in automotive systems, (b) Proposed solutions or extensions to safety standards and (c) any specific methodologies or life cycle phases tailored to ML integration in functional safety. The extracted information was then synthesized to identify common themes and gaps in the literature related to software requirements, data aspects, training processes, and model deployment.
5) **Gap Identification:** The systematic review revealed four key gaps, which are central to the contributions of this paper. Those gaps are in line with the gaps identified during the review of ISO 26262, for details see section II-B.

The SLR results were grouped into two categories: industry projects, discussed in section II-D1, addressing AI and ML safety in automotive applications, and research papers in section II-D2, which explore the integration of ML into the ISO 26262 framework.

1) INDUSTRY PROJECTS/INITIATIVES

Recent advancements in ensuring the safety of AI-based function modules for highly automated driving have been significant. Notable projects and standards have emerged to address the unique challenges posed by AI in automotive safety, specifically in the context of pedestrian detection and other critical functions. The integration of AI in automotive systems, particularly for highly automated driving, presents significant safety challenges. Various initiatives have been undertaken to address these challenges, focusing on developing robust safety frameworks and methodologies. However, notable gaps remain, which this paper aims to address by proposing a comprehensive approach tailored to the unique needs of ML-based systems in the automotive industry.

The *KI Absicherung* project [15] focussed on addressing the insufficiencies of existing safety processes for AI-based systems, specifically targeting the verification and validation of AI modules, such as pedestrian detection, in automated driving to ensure their safe integration into vehicles, which cannot be addressed using established safety processes. The project aimed to develop a joint safety argumentation methodology and foster industry consensus on safe AI. Over 36 months, this project involved various stakeholders, including OEMs, technology providers, research institutes, universities, and external partners, with a substantial budget of 41 million euros. However, the project highlighted the need for more comprehensive safety processes tailored specifically for AI technologies, which remain a gap in current practices. The Literature Repository KI Absicherung[1] provides a comprehensive collection of research and publications related to the KI Absicherung project. This repository includes valuable insights and methodologies that have been developed and can be referenced to understand the current state of AI safety research in automotive applications. By leveraging the findings and recommendations from these sources, we can better inform the development of extended life cycle phases and robust testing methods for ML models.

Leading automotive companies such as Bosch have been working on the AI safety landscape, developing methodologies to ensure the safety and reliability of AI in automated driving systems.[2] Bosch's efforts focus on creating safety frameworks for autonomous driving, but specific aspects of ML model deployment and training processes in automotive contexts are not comprehensively addressed, leaving room for further development as proposed in this paper.

Waymo's detailed Safety Framework for its autonomous vehicles emphasizes a multilayered approach to safety, including the hardware layer, ADS behavioral layer, and operations layer, each with specific safety verification and validation methods. Waymo's transparency in publishing its safety methodologies and performance data aims to foster trust and accountability in autonomous driving technology. However, like other initiatives, it does not fully cover the integration of ML-specific life cycle phases and rigorous testing methods proposed in this paper.[3]

CARIAD has been actively working on enabling the development of safety-critical AI functions through various initiatives such as *safety first for automated driving*[4] and contribution to development of standards for safety for automated driving systems such as [16]. Their involvement in AI safety research and the development of tools and processes underscores the growing industry focus on safe AI deployment in automotive systems.

Last, but not the least, a study on the application of AI in functional safety,[5] conducted by a panel of working groups across various industry sectors, identifies significant gaps in the current approach to AI safety in critical systems. The study highlights that traditional system development practices, which are deterministic in nature, are insufficient for AI-driven systems, which inherently involve

[1] https://www.ki-absicherung-projekt.de/en/news/news-detail/literature-repository-published
[2] https://www.bosch.com/stories/artificial-intelligence-in-cars/
[3] https://www.telematicswire.net/waymo-sharing-safety-framework-for-fully-autonomous-operations/
[4] https://group.mercedes-benz.com/documents/innovation/other/safety-first-for-automated-driving.pdf
[5] https://electrical.theiet.org/media/ifbjt25i/the-application-of-artificial-intelligence-in-functional-safety.pdf

non-deterministic components. Further, it identifies that the existing safety standards, such as IEC 61508 do not fully address unique AI challenges like bias and explainability, requiring the development of alternative safety arguments tailored to AI. Moreover, AI systems require additional activities during design, implementation, and testing stages, particularly regarding specialized verification and validation techniques that extend beyond traditional methodologies. Further, it identifies that ethical considerations, such as governance, trust, and fairness, emerge as crucial in the AI development process, and necessitate new frameworks to ensure responsible use. Understanding the behavior of AI systems, particularly in their interaction with human operators, is also identified as critical for ensuring safe operation. Finally, the study underscores the lack of consensus on managing the AI life cycle in safety-related applications, stressing the need for evolving standards and practices as AI's role in critical systems continues to grow.

Thus, an analysis of industry projects and initiatives reveals several key gaps in addressing the integration of ML into automotive safety frameworks. The identified gaps focus on main areas such as inadequate consideration of ML-specific software requirements, insufficient guidance on data management for ML models, limited coverage of the stochastic nature of ML training processes, and a lack of clear strategies for deploying ML models safely in embedded automotive systems. While projects like KI Absicherung and efforts by leading automotive companies have made progress, they do not fully address these challenges. Moreover, despite several other initiatives contributing to AI safety, significant gaps remain in adapting ML life cycle phases, testing methods, and deployment strategies within the ISO 26262 framework.

2) LITERATURE REVIEW

Apart from industry projects, the literature review focusing on recent research works on enhancing ISO 26262 with ML-specific life cycle phases and testing methods remains limited. The following discussion highlights key contributions in this area and summarizes the identified gaps.

The work presented in [12] discusses a novel concept for the systematic development of Deep Neural Networks (DNN) in automotive applications, addressing the gap in current standards like ISO 26262 and SOTIF, which do not fully accommodate DNN-specific characteristics. Their work focuses on developing systematic V-models for data management, the training process, and DNN integration into embedded hardware. These insights directly inspired our approach to extending ISO 26262 with ML-specific life cycle phases, particularly in addressing challenges around robustness, traceability, and training data uncertainties. This paper supports the need for new life cycle phases and standards, reinforcing our argument for an enhanced safety framework for ML-based systems in automotive safety-critical applications. Similarly, [17] investigates the integration of ML into automotive safety standards, particularly ISO 26262. The authors identify critical gaps in the standard regarding ML applications and suggest three key adaptations to bridge these gaps. However, the study primarily initiates the discussion and highlights the necessity for more comprehensive solutions. The work introduced by [18] presents a comprehensive assessment of ISO 26262 in the context of supervised ML and proposes modifications to better align the standard with the needs of ML-driven automotive systems. They highlight significant gaps in ISO 26262, particularly regarding the handling of non-deterministic behaviors inherent in ML models used for advanced driver assistance systems (ADAS) and automated driving systems (ADS). By systematically analyzing Part 6 of ISO 26262, the authors suggest new process requirements specifically designed to address these gaps. Their work underscores the critical need for an extended safety framework that accommodates ML-specific challenges, such as robustness, data management, and model verification, which directly align with the gaps identified in our research on software requirements and ML life cycle phases. This paper strengthens the case for adapting safety standards to ensure the safe deployment of ML in automotive systems.

In [19], the authors address key issues in ML safety such as interpretability, robustness, and verification of ML models in autonomous vehicles. The paper discusses gaps in ISO 26262 and provides algorithmic techniques to improve ML safety. It emphasizes the challenge of interpretability and transparency in ML models, which is crucial for safety-critical systems, directly linking to the gap in software requirements. While the authors successfully address key gaps such as interpretability, robustness, and error detection in ML models, they fall short in fully addressing the gap related to the life cycle management of AI systems in compliance with ISO 26262. Additionally, the authors in [20] survey techniques to build confidence in ML systems, highlighting the lack of a cohesive framework for integrating these methods into the ISO 26262 life cycle. Their findings on challenges like robustness, uncertainty, and interpretability align with the gaps identified in our research, particularly in software requirements, data aspects, and training processes. The absence of standardized guidelines for managing these ML-specific issues supports our call for extended life cycle phases and a more comprehensive framework within ISO 26262. This survey further underscores the need to address these gaps to ensure the safe deployment of ML in automotive systems.

The work discussed in [21] explores the potential of incorporating ML technologies in enhancing compliance with the ISO 26262 safety standard in automotive systems. One of the primary gaps identified in the paper is the lack of established norms and guidelines for applying ML and AI within the framework of ISO 26262. This echoes broader gaps in the application of AI to functional safety, where traditional methodologies for safety assurance struggle to account for the non-deterministic and probabilistic nature of AI systems. The paper also points to the stochastic

nature of ML algorithms, aligning with challenges identified in other research on interpretability and transparency in AI models. However, the paper falls short of providing practical methodologies for resolving these gaps. In [22], the authors propose a new approach to quantify ML failures in autonomous vehicles, treating them not only as systematic but also as random faults. The paper highlights challenges like training data uncertainties and the ''black box'' nature of ML models, aligning with the gaps identified in our research on software requirements and data aspects. Their focus on real-time failure rate assessment supports our call for enhanced life cycle phases and robustness metrics within ISO 26262, further validating the need for updated safety frameworks for ML-based systems in autonomous vehicles. Additionally, [19] discusses gaps in the ISO 26262 framework regarding the testing and validation of ML training processes, particularly in dealing with the stochastic nature of ML training, which introduces challenges not adequately covered by current standards.

In [23], the authors address the unique safety concerns posed by DNNs in automated driving systems, emphasizing the alignment of these concerns with standards like ISO 21448 (SOTIF) and ISO PAS 8800. Their work highlights the importance of adapting safety frameworks to manage DNN-specific issues such as uncertainty, robustness, and explainability, aligning with gaps in our research related to software requirements and model deployment. However, their approach primarily focuses on perception components, lacking a full life cycle framework for ML systems. The work presented in [24] proposes adapting ISO 26262 to address ML-specific challenges, focusing on dataset requirements and hazard analysis for autonomous driving. However, they do not fully address key gaps such as robustness, explainability, and the safe deployment of trained ML models. While the paper offers initial insights, further work is needed to systematically integrate ML-specific life cycle phases and testing methods.

Another recent work in [25] deals with a survey on AI for safety-critical systems in industrial and transportation domains. The paper identifies, among others, the need for standardization and certification of ML systems in alignment with ISO 26262 and the importance of integrating AI-specific methods into existing standards such as ISO 26262. [26] highlights the advancements in AI applications for automotive sectors like predictive maintenance, vehicle security, and Vehicle-to-Vehicle (V2V) communication, but it doesn't address some key areas. Notably, gaps related to ISO 26262, such as specific software requirements for AI/ML safety, the handling of training processes for robustness and uncertainty, and challenges in deploying AI models in embedded systems, remain unexplored.

In [27], the authors introduce *AI Safety Integrity Levels (AI-SILs)*, an extension of traditional SIL frameworks like ISO 26262. Their approach incorporates task complexity, using input entropy and output non-determinism, to better assess the risks posed by AI systems in safety-critical applications. This method addresses gaps in existing standards, which often fail to account for AI system complexities. By highlighting the need for more precise differentiation of AI risks, the paper supports our argument for extending ISO 26262 with AI-specific life cycle phases and risk assessments tailored to the unique challenges AI systems present. Similarly, [28] proposes methodologies to ensure the trustworthiness of high-risk AI systems in compliance with the European Union's AI Act. It introduces the Operational Design Domain (ODD) and Behavior Competencies (BC), borrowed from the automated driving domain, to assess risks across the AI life cycle. The paper emphasizes the role of standards and ethical considerations in shaping the certification of AI systems, aiming to foster human-centric and trustworthy AI. The methodology addresses gaps related to robustness, uncertainty handling, and interpretability but does not fully address life cycle phases specific to ML. Last, but not least, in [29], the authors conduct a comprehensive SLR of 329 key references, highlighting five main approaches for ensuring AI safety: black-box testing, safety envelopes, fail-safe AI, white-box explainable AI, and life cycle-based safety assurance. They emphasize the importance of integrating safety considerations throughout the life cycle of AI systems and propose future research areas such as dataset safety analysis and hyperparameter justification. This review aligns with our identified gaps in software requirements and verification processes, reinforcing the need for more structured approaches for AI safety assurance across industries.

The literature review highlights several key gaps in enhancing ISO 26262 for ML-based automotive systems. Current standards lack specific life cycle phases tailored to ML, particularly in areas like robustness, traceability, and systematic development [12], [17], [18]. Certification and validation guidelines remain underdeveloped, especially for handling the stochastic nature of ML algorithms and ensuring explainability [21], [27]. Critical ML safety properties such as robustness, uncertainty, and interpretability are not sufficiently addressed [19], [20], [30]. Further, there is limited guidance on managing ML training processes, particularly when using frameworks like TensorFlow or PyTorch [19], [22]. The standard also lacks structured processes for dataset validation and data management, which are crucial for ML model development [25]. Additionally, there is a gap in deploying ML models, especially when transitioning from high-level languages like Python to embedded C/C++ environments while maintaining safety properties [23]. Moreover, several works highlight the necessity of risk assessments and tailored life cycle phases for ML-specific systems [28], [29]. The AI Safety Integrity Levels (AI-SILs) framework is also proposed as an extension to address AI risks [27]. Last but not least, the comprehensive assessment of ISO 26262 in the context of supervised ML carried out in [18] and SLR in [29] provides a strong foundation and supports the need

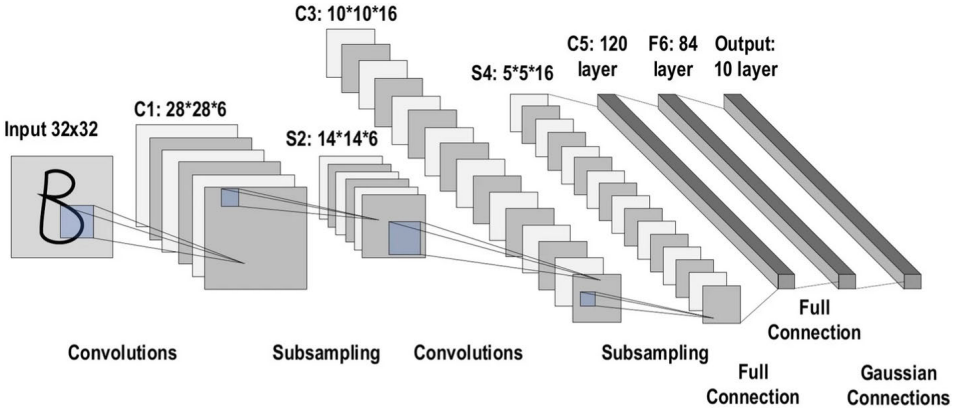

FIGURE 2. Example of a ML Unit [31].

for enhancing ISO 26262 to address ML-specific challenges, which is the focus of our work. Thus, these gaps underscore the need for a comprehensive integration of ML-specific life cycle phases, safety properties, testing methods and a rigorous evaluation framework to assess the effectiveness of these methods, into ISO 26262. Details regarding the specific properties of each life cycle phase are available in Section IV.

E. DEFINITIONS

In this section, we define key terms essential to our study's contributions and to addressing specific gaps in current standards. While we assume readers possess a foundational understanding of AI and ML-including familiarity with common terms, we provide explanations for specialized concepts such as *ML unit* and *ML-in-the-Loop*. These terms are pivotal to our work, particularly in the context of automotive systems and ML model deployment and testing.

1) MACHINE LEARNING (ML) MODEL

A mathematical construct that generates an inference based on input data or information and comprises functionality that is created by machine learning (cf. section 3.1.32 in [2]).

2) ML DATA MODEL

A ML Data Model consists of data subsets used for different purposes in the context of ML like training, validation and test datasets (cf. section III-B1 / 3.2.2 / 3.2.8 in [2]).

3) TRAINED ML MODEL

A ML model with a set of model parameters as a result of model training (cf. section 3.1.40 in [2]).

4) DEPLOYED ML MODEL

A ML model that has been optimized after the training process and deployed into a format compatible with the target platform, like embedded hardware. This corresponds to the interpretation of the deployed ML model given by ISO 8800 [2] and ASPICE 4.0 [32], although an explicit definition is not mentioned there.

5) ML-IN-THE-LOOP

Similar to the HW-in-the-Loop testing concept from ISO/IEC TR 5469 [3], the ML-in-the-loop testing concept enables testing of a deployed ML model in a real-operating environment. The ML model as well as the direct pre- and post-processing steps should be implemented on the target hardware to enable testing in an operation-like environment.

6) ML UNIT

A ML Unit is the smallest entity of code that encapsulates the essential components required to generate predictions. It includes only the minimal code necessary to process input data and produce output predictions, excluding the code used to load the model or data. For the example of (feed-forward) neural networks, the ML Unit is composed of various layers, interconnections between them, which all together build up the desired functionality. This definition is designed to be flexible, allowing the ML Unit to encompass future ML architectures and hardware developments. Please note that in the context of ASPICE 4.0 [32] an ML Unit can be compared with ML Element and ML Architecture, which similarly encapsulates the essential components of an AI-based function, but it is not limited by specific programming paradigms or hardware requirements. Figure 2 shows a typical neural network architecture of an image classification task. For products in the automotive industry, typical image classification tasks are traffic sign or traffic light classification. For the example architecture in Fig. 2, all elements in the figure would be considered part of the ML Unit. The final work product of ML development is an embedded ML function, which runs on the target hardware. The embedded ML function, and therefore the ML Unit, consists of:

- Diverse layers, which are given as e.g. C/C++ code,
- Weights, which are e.g. concrete integer 8-Bit values.

For the example of neural networks, (SW) layers of an ML function might be interpreted as SW units regarding ISO 26262-6. Because of the limited scope of ISO 26262 and gaps related to ML technology, a new test level for an ML Unit shall be introduced and defined. The definition includes not

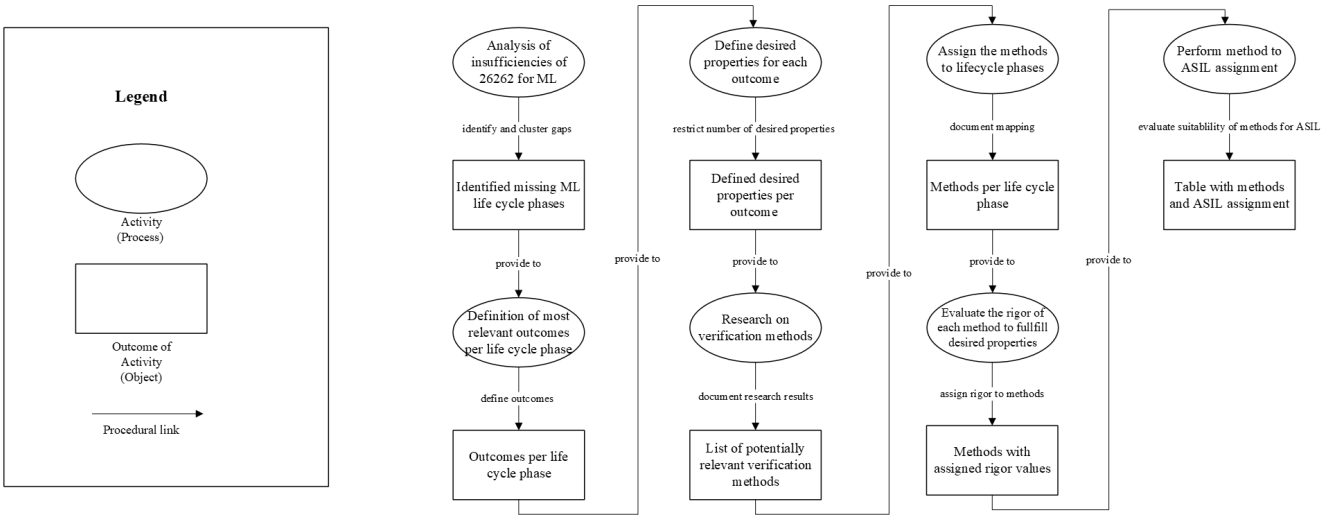

FIGURE 3. Steps involved in the development of a systematic ML testing concept for ASIL A/B conformance.

the single (SW) layer of an ML function, but the overall ML model with all parts that are directly related to the task of the ML model (e.g. layers and weights). The definition of an ML Unit will allow to:

- Address ML specific aspects that are not covered by ISO 26262,
- Perform requirements-based and interface tests,
- Introduce new testing methods to prove ML specific safety properties (e.g. robustness, uncertainty and interpretability).

III. SYSTEMATIC APPROACH

Fig. 3 illustrates our applied approach for the definition of the novel, systematic ML testing approach for ASIL A/B conformance introduced in this paper. Since expert judgement was an important aspect in performing the steps mentioned in Fig. 3, this section provides further insights into the expert judgement performed. The approach is inspired by the 61508 (Part 3, Annex C), which makes use of the so-called ''desired properties'' and ''rigor''. This section of the paper provides more details on how those terms have actually been used by the authors in developing the concept and can be split into the following steps;

- Identifying missing, ML-specific life cycle phases
- Defining desired (safety) properties for the outcomes of each life cycle phase
- Designing test methods that support the achievement of the desired properties
- Evaluating the test methods for the desired properties
- Determining the test methods' rigor
- Assigning test methods to ASIL

A. EXPERT JUDGMENT PROCESS

To ensure the scientific rigor and validity of our study, a structured expert judgment process was employed. This multi-step evaluation involved professionals (including the authors of this paper) with expertise in classic functional safety and AI standards integration and AI Auditing[6,7,8]. The process aimed to systematically assess the proposal's applicability in the context of safety-critical AI systems, ensuring alignment with relevant industry standards.

- **Expert Selection and Preparation:** Experts were selected based on their extensive experience in AI auditing, functional safety, and standards such as ISO 29119 and ISO PAS 8800. Their collective expertise provided a solid foundation for evaluating the proposal from multiple perspectives, ensuring a comprehensive review.
- **Evaluation:** An initial proposal of the study was created by the authors of this paper, following the steps outlined in Fig. 3. This proposal was subject to evaluation by experts. The experts were tasked with evaluating the proposal's alignment with existing safety standards, particularly ISO 26262, as applied to AI/ML applications. This focused on several key areas, including robustness, uncertainty handling, and identifying gaps in current safety frameworks.
- **Questions Asked:** Experts were asked to assess the proposal's alignment with safety standards and the adequacy of the testing methods and desired properties. Key questions included (a) do the selected test methods comprehensively address the identified safety properties?, (b) are the proposed life cycle phases and testing methods appropriate for ML safety assurance? and (c) how effectively does the proposal handle various desired properties?
- **Structured Feedback and Collaborative Assessment:** Using methodologies similar to those outlined by [33] and [34], the experts engaged in a structured, col-

[6]https://innotecsafety.com/
[7]https://cariad.technology/
[8]https://www.trustifai.at/

laborative evaluation process. This process involved discussions that highlighted inconsistencies across standards, identified gaps in current practices, and provided recommendations for improvement based on empirical insights. These insights were derived from the experts' practical experience and real-world applications ensuring that the recommendations were grounded in evidence from actual implementations in safety-critical AI systems.

- **Data Collection and Analysis:** Expert feedback was systematically gathered, documented, and analyzed using various qualitative techniques. The method used was the use of an Excel sheet to collect, categorize, and track review comments from evaluation, enabling systematic revisions and ensuring that all feedback was addressed. This method ensured a comprehensive review process, allowing for the iterative refinement of the proposal by addressing identified gaps. Feedback was assessed qualitatively across two rounds of discussions. The first round included four experts, and the second included six. While no formal statistical tests were conducted, the consistency of expert feedback was noted, and qualitative agreement was achieved.
- **Synthesis and Application of Feedback:** The expert feedback was synthesized into recommendations, which were incorporated into the proposal to improve its overall robustness and ensure better alignment with relevant industry standards.

By applying this method of expert judgment, we have derived conclusions that are documented in our approach. The main benefit of our approach is the thorough development of the concept. The second benefit is that we systematically set up the expert group which supports the maturity of the results.

B. DEFINITION OF LIFE CYCLE PHASES

In comparison of the life cycle phases of ISO 26262 with the activities of ML product development, three major gaps can be highlighted:

- Prepare data
- Train ML model
- Deploy ML model

In this paper, these life cycle phases are referred to as *ML life cycle phases*, as illustrated on the bottom side of Fig. 4. As an extension to Fig. 4, Fig. 6 in Appendix A provides an elaboration on how these life cycle phases are embedded into the traditional V-Model and aligned with the life cycle phases of ISO 26262.

To perform these life cycle phases, artifacts (e.g. inputs and outputs of each life cycle phase) as defined by ISO 26262 need to be provided as input. It needs to be specified upfront, what the requirements are for the functionality of the ML model. These requirements need to cover the functionality of the ML model as well as the syntactic and semantic requirements for the data, and should be derived following the requirements engineering life cycle of ISO 26262. Necessary adaptations of this life cycle phase to cover,

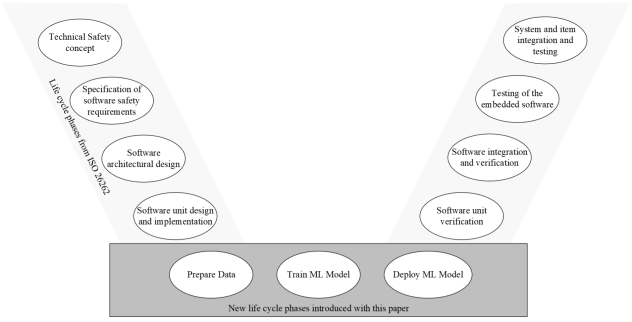

FIGURE 4. Interfaces to existing life cycle phases of ISO 26262-Part 6.

for example, the specification of data requirements are not part of this document.

In addition to the requirements, a detailed design specification as defined by ISO 26262 is a relevant input to the life cycle phases of this document. The deployed ML model as output of the life cycle phase *deploy ML Model* is considered to be an AI Unit, which needs to be specified with other SW Units in the detailed design specification. Information such as specified interfaces to other SW Units needs to be considered in the life cycle phase *deploy ML model*, which makes the detailed design specification a mandatory input to this development step.

1) DATA HANDLING

There is an important separation between ML-specific data handling and ML-agnostic data handling. This document only focuses on ML-specific data handling. The data life cycle phase that is part of this document requires quality-assured data and metadata as input, which is not defined in the existing life cycle phases of ISO 26262. Aspects such as data acquisition, data storage, and data labeling/annotating are not considered specific to ML since these activities are also relevant for verification and validation purposes in non-ML products. These activities are crucial for ensuring the quality of the data, and we assume that measures are in place to maintain this quality, although they are not described in detail in this paper. These quality assurance measures might be derived from performing a Process Failure Mode and Effects Analysis (P-FMEA) [35] on the aforementioned aspects—acquisition, storage, and annotation/labeling—as recommended by the Safety of the Intended Functionality (SOTIF) standard [5].

For example, the following steps might be undertaken to ensure data quality:

- Reviewing the requirements for the target data acquisition to ensure they are comprehensive and meet the necessary standards.
- Verifying that the hardware used for capturing the data is configured correctly and operates reliably.
- Performing tool classification and qualification of the labeling tools to ensure they are suitable for the task and produce consistent, high-quality results.

These measures are assumed to be in place to provide a solid foundation for the ML-specific processes described in this paper.

2) TRAINING OF ML MODEL

The training of an ML model is part of the second life cycle phase of the approach outlined in this paper. For many applications, the outcome of the training process will not be the product that is used during operations. Therefore, this paper provides a third life cycle phase that describes the deployment of the ML models after training. For those applications, the quality of the ported or deployed version of the ML model is the most critical, since this version will be used during operations. Due to this criticality, the testing activities on the deployed ML model should be prioritized over the testing of the trained ML model. We assume that a systematic development and testing approach for the trained ML model also supports the quality of the deployed ML model and the identification of systematic issues, and therefore demands the application of test methods on the trained ML model.

3) DEPLOYMENT OF ML MODEL

The third life cycle phase (model deployment) consists of the test methods of the second life cycle phase (model training) and extends those with deployment-specific test methods (e.g., related to hardware restrictions that are not relevant to the trained model). Since the deployed ML model is considered as a ML Unit, integration steps with other SW Units and SW Components are necessary before the deployed ML model can be used in operations. At this point, the interface to the existing life cycle phases of ISO 26262 comes into play again. The interfaces to the existing life cycle phases of ISO 26262, as described above, are also illustrated in Fig. 4 and Fig. 6.

C. DEFINITION OF DESIRED PROPERTIES

For each life cycle phase, there is a significant outcome artifact that serves as the foundation for subsequent phases. For example, the trained ML model is a key input for the *deploy ML model* life cycle phase. Each of these major outcomes has clearly defined desired properties. These desired properties represent the essential nonfunctional requirements that should be prioritized during development and testing to ensure high-quality results. Examples of these desired properties are completeness and consistency. The desired properties outlined in this paper focus on the most critical aspects, carefully selected based on expert judgment. This targeted approach ensures that the most relevant and impactful properties are emphasized, facilitating a thorough and effective evaluation process. By concentrating on key properties such as robustness, explainability, and uncertainty handling, this framework aims to improve the current practices in ML model development and testing, addressing gaps in existing standards. The purpose of this approach is to lay the groundwork for systematic testing of ML models, contributing to the development of more reliable and safe automotive systems.

D. DEFINITION OF TEST METHODS

There are several standards that provide robust test methods for developing and testing ML models, each contributing valuable insights and methodologies. For this paper, we considered ISO 29119 (Part 11) and ISO PAS 8800. The selection of both the standards and the test methods was based on expert judgment, acknowledging that many of these methods are complementary, which enhances their effectiveness in ensuring safety.

In the initial step, we thoroughly analyzed the test methods in these standards for their applicability to the ML life cycle phases. This analysis resulted in a detailed mapping of test methods to the specific phases of the ML life cycle. For each test method, we developed a concise definition and provided a concrete and pragmatic description to ensure clarity and ease of implementation.

It is important to emphasize that while specification and design are crucial activities in the life cycle phases discussed in this paper, our primary focus is on testing methods. These methods are essential for ensuring the robustness, reliability, and safety of ML models. By leveraging the comprehensive guidelines and methodologies from ISO 29119-11 and ISO PAS 8800, we aim to establish a solid foundation for the systematic testing of ML models, ultimately contributing to the advancement of safety-critical automotive systems.

E. EVALUATION OF TEST METHODS, RIGOR DERIVATION AND ASIL ASSIGNMENT

In this step, all test methods that have been mapped to the ML life cycle phases are evaluated with respect to their effectiveness in supporting the achievement of the desired properties. A single test method can support the achievement of several desired properties; we assumed that there is a main and potentially several minor desired properties that are supported by a test method.

1) RATIONALE BEHIND RIGOR ASSIGNMENT

In the first step, the test methods have been mapped to the desired properties they support in general, and then the major desired property for the test method has been defined. The degree to which a desired property is supported by the test method is defined with a rigor, similar to IEC 61508 (cf. Fig. 5). For this paper, there was a separation between Rigor Level 1 (R1) and Rigor Level 2 (R2). The rationale behind the assignment of R1 and R2 for each test method and desired property is based on the effectiveness and thoroughness of the test methods in addressing the specific desired properties.

R1 (Rigor Level 1): Indicates that the method is effective and suitable for safety-related aspects with lower criticality, such as those classified under ASIL A in functional safety standards like ISO 26262. Methods assigned R1 provide adequate assurance for functions where safety measures are necessary but the required rigor is moderate.

R2 (Rigor Level 2): Indicates that the method is highly effective and suitable for critical aspects where thoroughness is essential, such as those classified higher than ASIL A in functional safety standards like ISO 26262. Methods assigned R2 offer a higher level of confidence in achieving the desired properties and are often more comprehensive and resource-intensive.

Please note that, on a fine granular level, test methods can be executed in several ways. Therefore, two test method variants have been defined for each test method. The test method variants vary in their effectiveness to support a desired property.

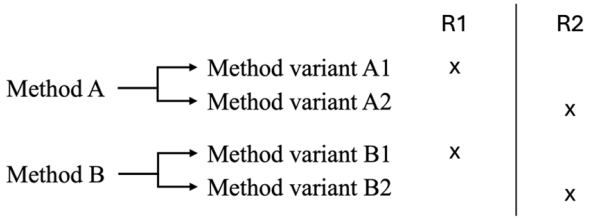

FIGURE 5. Rigor assignment.

After the rigor R1 and R2 have been assigned to the test method variants, recommendations for the test method application have been derived, considering the ASIL. In general, the paper differentiates between ASIL A and ASIL B. For ASIL A, all test methods that have an R1 assigned are highly recommended (HR), and the test methods with R2 are recommended (R). For ASIL B, all test methods that have R2 are HR, and the test methods with R1 are R.

IV. EVALUATION AND RESULTS FOR ML DATA MODEL

In this section, the systematic approach defined in section III is applied to the first ML product life cycle phase, namely *prepare data*. Similar to ISO 26262, the objectives and inputs of the respective life cycle phase are defined. This is followed by a definition of the set of desired safety properties for the prepare data life cycle phase and the selection of test methods for this phase. The mapping of desired properties and test methods for the ML data model is subsequently outlined. Following the rigor assignment, a mapping of the ASIL level (A or B) and test methods for this life cycle phase is described. Thus, the presented approach provides a comprehensive framework for evaluating and ensuring the safety of ML data models in automotive systems. This framework includes:

- **Objectives:** Clearly defined goals to ensure the data model meets operational and safety requirements.
- **Inputs:** Necessary data and metadata, system requirements, and architectural specifications.
- **Desired Properties:** Identification of key safety properties such as completeness, consistency, correctness, representativeness, and independence.
- **Test Methods:** Selection of appropriate methods to verify the data model.
- **Mapping of Properties to Methods:** Aligning desired properties with specific test methods to ensure a comprehensive evaluation.
- **Rigor Assignment:** Assigning rigor levels (R1 and R2) to test methods based on their effectiveness in achieving desired properties.
- **ASIL Mapping:** Recommendations for test methods based on ASIL A or B to guide their application in safety-critical contexts.

By following this structured approach, the framework not only enhances the robustness and reliability of ML data models but also aligns with established safety standards, facilitating their integration into automotive systems.

The purpose of the prepare data life cycle phase is to enable the development of data-driven, safety-critical, and quality-managed products by providing verified datasets for ML model training, validation, and testing.

A. OBJECTIVES

- Provide evidence that the implemented ML data model fulfills the allocated Operational Design Domain (ODD) to enable the fulfillment of the system and software requirements.
- Verify that the defined safety measures resulting from safety-oriented analyses are properly implemented.

B. INPUTS

- Labeled data and metadata that achieved the safety goals/quality criteria.
- ODD definition, part of system requirements.
- Functional/logical architecture of the system.
- Functional/logical architecture of the software.

C. DESIRED PROPERTIES FOR THE LIFE CYCLE PHASE PREPARE DATA

In the following, the safety desired properties for the first life cycle phase, namely prepare data, are discussed. Each property is introduced with a brief description and a simple example. The examples, based on a Traffic Sign Recognition (TSR) classifier use case, provide *representative scenarios* demonstrating how these properties apply in practice.

- **Completeness:** The goal is to ensure that the data comprehensively covers all relevant aspects of the problem domain, including temporal aspects. This involves capturing all necessary elements, attributes, and relationships to meet the system and software requirements (ODD). Completeness requires the dataset to include all necessary elements and scenarios that the ML model will encounter in real-world applications, ensuring it is thorough and accounts for all variations in the ODD, including a wide range of features, variations, and edge cases relevant to the intended application domain.
  Example: For a TSR classifier, completeness would mean having a dataset that includes images of all

possible traffic signs under various conditions such as different lighting, weather, and occlusions.

- **Consistency:** The objective is to maintain consistency within the data to avoid contradictory or conflicting information. Consistency in the data ensures uniformity and coherence in the representation and interpretation across different components. It maintains alignment with the system requirements (ODD) and the functional/logical architecture, facilitating accurate and reliable inference and decision-making. This means that data should be represented in a coherent and standardized format, avoiding discrepancies that might confuse the ML model.
  Example: For a TSR classifier, consistency would entail ensuring that all traffic sign images are captured in similar resolutions and formats, with annotations following the same labeling conventions.
- **Correctness:** The aim here is to verify the correctness of data to minimize errors and inaccuracies during model training and evaluation. The data should correspond to the phenomenon they intend to capture, including features and metadata that help characterize the phenomenon [2]. This involves checking for errors, mislabeling, and inaccuracies within the dataset.
  Example: For a TSR classifier, correctness would require that each image is correctly labeled with the right traffic sign and that there are no mislabeled or incorrect entries in the dataset.
- **Representativeness:** The goal here is to guarantee that the data accurately represents the underlying distribution of real-world data to avoid bias and ensure generalization. This property is crucial for training models that generalize well to unseen data. Representativeness ensures that the data adequately reflects the characteristics, structures, and relationships inherent in the real-world domain it represents. It involves capturing a diverse and representative sample of data that accurately represents the variability and complexity of the operational environment, supporting robust AI model training, validation, and testing.
  Example: For a TSR classifier, representativeness would involve having a dataset with traffic sign images from various geographical locations, different road types, and varied environmental conditions.
- **Independence of datasets** The objective here is that the datasets sufficiently avoid leakage of information amongst themselves with respect to data sources and the methods used to capture, gather, generate, and process the data. By ensuring the independence of data sets, one can avoid leakage of information between the three datasets (training, validation, testing). In other words, independence refers to the necessity for data samples to be distinct and not overly redundant. This property helps in avoiding overfitting and ensures that the model learns to generalize rather than memorize.
  Example: For a TSR classifier, independence would mean ensuring that the same samples aren't contained in different datasets, which could lead to the model overfitting to those specific examples.

D. TEST METHODS FOR THE LIFE CYCLE PHASE PREPARE DATA

This section presents a list of test methods for the life cycle phase prepare data. Each test method includes two variants, providing flexibility and comprehensiveness to address different testing requirements effectively.

- **Consistency Testing:** Ensures that the data used for training, validation, and testing is consistent in format and content.
  - *Random Data Consistency Testing:* Randomly selects data entries to check for consistency. This involves checking random samples of data entries to ensure they meet the predefined consistency criteria, e.g. verify that they conform to predefined formats and standards. For example, in a TSR classifier, this would mean randomly checking images to ensure they have consistent resolutions and correct labeling.
  - *Systematic Data Consistency Testing:* Systematically checks all data entries for consistency. This method involves a thorough and systematic review of the entire dataset to ensure all entries are consistent with each other and the defined standards. In the TSR example, this would involve systematically verifying that all images meet the necessary format and labeling standards across the entire dataset.
- **Distribution Analysis:** Analyzes the distribution of data to ensure it is representative of the target environment.
  - *Key Features Data Distribution Analysis:* Focuses on key features to ensure their distribution is representative. This method analyzes the distribution of key attributes in the dataset to ensure they match the expected distribution patterns. For a TSR classifier, this would mean ensuring that the dataset has a balanced distribution of traffic sign types (e.g., stop signs, yield signs) under various conditions.
  - *All Features Data Distribution Analysis:* Analyzes the distribution of all features to ensure overall representativeness. This comprehensive approach checks the distribution of all attributes to confirm they are representative of the operational design domain. In the TSR example, this would include verifying that images are representative of different times of day, weather conditions, and levels of occlusion
- **Data Augmentation:** Enhances the dataset by adding modified versions of existing data to increase diversity.
  - *Random Data Augmentation:* Randomly augments data entries to increase variability. This involves

randomly modifying data samples to create new, varied examples. For example, in a TSR classifier, this could involve randomly applying transformations such as rotation, scaling, and color adjustments to traffic sign images.
  - *Systematic Data Augmentation:* Systematically augments data entries based on specific rules. This method applies predefined rules to systematically create new data samples from existing ones. In the TSR use case, this might involve systematically generating variations of traffic sign images to cover different scenarios like lighting changes and weather conditions

- **Representativeness Testing:** Ensures the data represents the operational design domain accurately.
  - *Representativeness Testing Key Attributes:* Focuses on key attributes to ensure representativeness. This method checks whether key attributes in the dataset accurately reflect the target environment. For a TSR classifier, this would involve testing to ensure that the dataset includes a representative sample of all critical traffic sign types.
  - *Representativeness Testing All Attributes:* Tests all attributes to ensure comprehensive representativeness. This involves evaluating all attributes in the dataset to ensure they represent the operational design domain. In the TSR example, this would include verifying that the dataset accurately reflects all relevant environmental conditions and scenarios
- **Independence Testing:** Checks that data samples are independent and not duplicates or overly similar.
  - *Independence Test on Duplicates:* Identifies and removes duplicate data entries. This method ensures no duplicate entries exist in the dataset, maintaining data integrity. For a TSR classifier, this would involve checking for and removing identical traffic sign images.
  - *Independence Test with Similarities:* Ensures data entries are sufficiently different from each other. This involves checking data samples to ensure they are independent and not overly similar, preserving the diversity of the dataset. In the TSR use case, this would mean verifying that traffic sign images are not overly similar, preventing redundancy in the dataset.
- **Review:** Involves stakeholders in reviewing the data and methods to ensure completeness and correctness.
  - *Review with Some Stakeholders:* Involves a subset of stakeholders in the review process. This method includes selected stakeholders in the review to validate the data and methods. For a TSR classifier, this might include experts from the development team reviewing the dataset for accuracy and completeness.
  - *Review with All Stakeholders:* Involves all relevant stakeholders in the review process. This comprehensive review process ensures all relevant parties validate the data and methods. In the TSR example, this could involve a review by the entire project team, including developers, testers, and domain experts.

E. MAPPING OF DESIRED PROPERTIES, METHODS FOR AI DATA MODEL AND RIGOR ASSIGNMENT

The desired properties for the AI data model outlined in section IV-C are mapped to specific test methods mentioned in section IV-D to ensure they are effectively achieved and presented in Table 1. The methods are evaluated for their ability to support each desired property, with rigor levels assigned to indicate their effectiveness. Please note that the general rationale behind rigor assignment (i.e., R1 or R2) is outlined in section III-E. In this section, we detail how each method supports the desired properties.

Table 1 and the descriptions above illustrate how each method supports key desired safety properties such as completeness, consistency, correctness, representativeness, and independence for the ML data model. The rigor levels (R1 and R2) indicate the method's effectiveness, with R2 representing a higher level of rigor.

F. MAPPING OF ASIL LEVEL AND TEST METHODS FOR ML DATA MODEL

The assignment of ASIL levels to test methods is based on the rigor and criticality of each method. The assignment of ASIL levels (A or B) follows this rationale, and the overall approach is based on expert judgment. Experts assess the methods by considering factors such as the criticality of the function, comprehensiveness, and confidence required to meet safety standards. For ASIL A, methods that provide sufficient assurance for non-critical functions are highly recommended (HR) if they are moderately rigorous (R1). For more critical aspects requiring higher confidence, the methods are recommended (R). Conversely, for ASIL B, methods assigned with high rigor (R2) are highly recommended (HR), while those with moderate rigor (R1) are simply recommended (R).

Table 2 presents the mapping of ASIL to the various test methods used for the life cycle phase prepare data. This table details how each method and its variants are recommended for different ASIL levels, specifically ASIL A and ASIL B. A variety of methods such as consistency testing, distribution analysis, data augmentation, representativeness testing, independence testing, and review processes are provided in the table. For each method, Table 2 indicates whether it is highly recommended (HR) or recommended (R) for ASIL A and ASIL B, helping guide the application of these methods based on the required safety integrity level.

V. EVALUATION AND RESULTS FOR TRAINED ML MODEL

In this section, the systematic approach described in section III is applied to the second ML product life cycle phase,

TABLE 1. Mapping of desired properties and methods for ML data model.

Method	Method Variant	Complete-ness	Consist-ency	Correct-ness	Represen-tativeness	Indepen-dence
Consistency Testing	Random Data Consistency Testing		R1			
	Systematic Data Consistency Testing		R2			
Distribution Analysis	Key Features Data Distribution Analysis	R1			R1	
	All Features Data Distribution Analysis	R1			R2	
Data Augmentation	Random Data Augmentation	R1		R1	R1	
	Systematic Data Augmentation	R1		R2	R1	
Representativeness Testing	Representativeness Testing Key Attributes	R1			R1	
	Representativeness Testing All Attributes	R2			R2	
Independence Testing	Independence Test on Duplicates					R1
	Independence Test with Similarities					R2
Review	Review with Some Stakeholders	R1	R1	R1	R1	R1
	Review with All Stakeholders	R1	R2	R2	R1	R1

TABLE 2. Test methods for the life cycle phase prepare data with ASIL mapping.

Method	Method Variant	ASIL A	ASIL B
Consistency Testing			
	Random Data Consistency Testing	HR	R
	Systematic Data Consistency Testing	R	HR
Distribution Analysis			
	Key Features Data Distribution Analysis	HR	R
	All Features Data Distribution Analysis	R	HR
Data Augmentation			
	Random Data Augmentation	HR	R
	Systematic Data Augmentation	R	HR
Representativeness Testing			
	Representativeness Testing Key Attributes	HR	R
	Representativeness Testing All Attributes	R	HR
Independence Testing			
	Independence Test on Duplicates	HR	R
	Independence Test with Similarities	R	HR
Review			
	Review with Some Stakeholders	HR	R
	Review with All Stakeholders	R	HR

namely *train ML model*. Similar to ISO 26262, the objectives and inputs of the respective life cycle phase are defined. This is followed by a definition of a set of desired properties for the train ML model life cycle phase, and then a selection of test methods for this phase. The mapping of desired properties and test methods for the trained ML model is subsequently outlined. Following the rigor assignment, a mapping of the ASIL level (A or B) and test methods for this life cycle phase is described. Thus, the presented approach provides a comprehensive framework for evaluating and ensuring the safety of trained ML models in automotive systems. This framework includes:

- **Objectives:** Clearly defined goals to ensure the trained model meets operational and safety requirements.
- **Inputs:** Necessary data and metadata, system requirements, and architectural specifications.
- **Desired Properties:** Identification of key safety properties such as functional correctness, accuracy, robustness, explainability, and uncertainty handling.
- **Test Methods:** Selection of appropriate methods to verify the trained model.
- **Mapping of Properties to Methods:** Aligning desired properties with specific test methods to ensure a comprehensive evaluation.
- **Rigor Assignment:** Assigning rigor levels (R1 and R2) to test methods based on their effectiveness in achieving desired properties.
- **ASIL Mapping:** Recommendations for test methods based on ASIL (A or B) to guide their application in safety-critical contexts.

The purpose of the life cycle phase train ML model is to enable the development of data-driven, safety-critical, and quality-managed products by providing a use-case-specific ML model that has learned the required functionality and is suitable to be deployed to the target platform.

A. OBJECTIVES

- Ensure the learning progress of the required functionality (specified in the software requirements) from data (ML data model).
- Verify that the learned functionality/trained model fulfills the allocated software requirements according to the required ASIL.
- Ensure that the defined safety measures resulting from safety-oriented analyses are properly implemented.

B. INPUTS

- ML data model (in accordance with the prepare data life cycle phase).

- Software requirements (including ML specifics).
- Software unit design specification.

C. DESIRED PROPERTIES FOR THE LIFE CYCLE PHASE TRAIN ML MODEL

The following discusses the safety desired properties for the identified second life cycle phase, namely train ML model, are discussed. Each property is introduced with a description and a simple example. The examples, based on a TSR classifier use case, provide *representative scenarios* demonstrating how these properties apply in practice.

- **Functional Correctness:** The aim here is to ensure that the trained model (e.g. Neural Network (NN)) correctly implements the desired functionality and produces accurate predictions. Functional correctness refers to the ability of the trained NN model to accurately learn and perform the required functionality specified in the software requirements. It ensures that the model behaves as intended and produces correct outputs for given inputs, meeting the functional expectations of the system.
  Example: For a TSR classifier, functional correctness would mean that the model accurately identifies and classifies traffic signs in various images, correctly recognizing the sign type and any associated actions.
- **Accuracy:** The objective here is to make sure that the model produces accurate predictions or classifications on unseen data, meeting specified performance metrics. Accuracy in the context of the NN model pertains to the model's ability to produce correct and reliable predictions or classifications for input data. It involves verifying that the learned functionality or trained model achieves a high level of accuracy in fulfilling the allocated software requirements, as well as meeting the required ASIL level.
  Example: For a TSR classifier, accuracy would be measured by the percentage of traffic signs correctly classified out of the total number of signs in the test dataset.
- **Robustness:** The goal here is to design the model to be robust against variations in input data and perturbations, enhancing its resilience in diverse environments. The robustness of the NN model refers to its ability to maintain performance and reliability across diverse and challenging conditions, including variations in input data, environmental factors, and operational scenarios. A robust model can generalize well to unseen data and handle uncertainties or perturbations without significant degradation in performance.
  Example: For a TSR classifier, robustness would involve testing the model's ability to correctly recognize traffic signs under varying lighting conditions, partial occlusions, and different weather scenarios.
- **Explainability:** The aim here is to develop models that are explainable and transparent, enabling one to understand the reasoning behind predictions and decisions. Explainability refers to the transparency of the NN model's decisions and predictions. It involves ensuring that the model's internal workings are understandable and interpretable, enabling to assess its behavior, identify potential biases or errors, and make informed decisions based on its outputs [2].
  Example: For a TSR classifier, explainability would involve providing insights into why the model classified a particular image as a stop sign, potentially highlighting the features or patterns that influenced the decision.
- **Uncertainty Handling:** The aim here is to implement safety mechanisms to handle uncertainty to mitigate the impact of erroneous predictions, especially in critical applications. Uncertainty handling involves the model's ability to recognize and communicate the confidence or uncertainty associated with its predictions. This is important for understanding the reliability of the model's outputs.
  Example: For a TSR classifier, uncertainty handling would involve the model providing confidence scores for each classification, indicating how certain it is about the identification of each traffic sign.
- **Generalization:** The aim here is to ensure that the model is able to generalize well to unseen data or scenarios beyond the training dataset, demonstrating its ability to capture underlying patterns and make reliable predictions beyond the training dataset. It involves assessing the model's capacity to learn meaningful representations from the data and apply them effectively to new inputs, ensuring robust performance in real-world applications.
  Example: For a TSR classifier, generalization can be evaluated by testing the model on images of traffic signs not included in the training dataset. This includes signs that differ in some semantic dimensions, but within the same classes. The TSR classifier should accurately recognize and classify these unseen signs, demonstrating its ability to generalize learned patterns to new inputs.

D. TEST METHODS FOR THE LIFE CYCLE PHASE TRAIN ML MODEL

This section outlines the test methods selected for the train ML model life cycle phase. Each method is carefully chosen to ensure the model meets functional correctness, accuracy, robustness, explainability, and uncertainty handling requirements. The methods are categorized and detailed to provide a comprehensive framework for evaluating the trained model.

The selection of these methods corresponds with guidelines provided in *ISO 29119*, Chapters 7, 8, and 9 [6]. Specifically, Chapter 7 covers Testing and QA of ML systems, including adversarial examples, benchmarks, and hyperparameter optimization (Sections VIII and 7.9). Chapter 8 emphasizes black-box testing approaches, including

combinatorial and metamorphic testing (Sections 8.1 and 8.4), while Chapter 9 focuses on white-box testing for neural networks, particularly neuron coverage (Section 9.2.2). Additionally, *Annex A.8* in [6] provides guidance on machine learning performance metrics, which is highly relevant for assessing the overall performance of ML models.

- **Requirements-Based Testing:** Ensures that the ML model adheres to the specified software requirements.
  - *No variants:* This involves verifying that the trained model meets all the requirements specified in the software design documents. For example, in a TSR classifier, this would mean ensuring that the model accurately detects and classifies traffic signs as specified in the requirements (e.g. classification accuracy 99 percent, no false positives).
- **Initial Performance Evaluation:** Assesses the performance of the ML model to ensure it meets initial performance criteria.

  Specific tasks could include:
  - ∗ Evaluating the accuracy of the neural network model in correctly classifying input data samples into their respective categories or classes.
  - ∗ Measuring the proportion of correctly predicted outcomes compared to the total number of test samples.
  - ∗ Evaluating the model's performance against ground truth labels or expected outputs to measure its precision and correctness.
  - ∗ Conducting quantitative analysis of accuracy metrics such as classification accuracy, precision, recall, and F1 score to assess model performance.
  - ∗ Conducting accuracy testing using a separate validation and test dataset or through cross-validation techniques to assess the model's generalization ability and performance under unseen data.
  - *Without Evaluations on Data Subsets:* Evaluates the model's performance on the entire test dataset to ensure overall functionality. This involves measuring the accuracy and other metrics of the model's predictions to create a performance baseline for other tests like robustness testing. This method evaluates the model's overall performance without breaking down the test dataset into subsets. For a TSR classifier, this would involve assessing the model's accuracy across the entire test dataset.
  - *With Evaluations on Data Subsets:* Evaluates the model's performance on specific subsets of the test dataset to identify performance variations across different data segments. This approach allows for a more detailed analysis of how the model performs under various conditions and data distributions. In the TSR use case, this might involve testing the model separately on subsets of the test dataset representing different weather conditions or times of day.
- **Robustness Testing:** Ensures that the ML model performs reliably under various conditions. This method evaluates the model's performance under different perturbations and edge cases to assess its robustness, ensuring it can handle reasonably expected conditions and environments.
  - *Simple Input Perturbations:* This method tests the model's robustness by introducing simple perturbations to the input data like noise or perturbations within the ODD. For example, in a TSR classifier, this could involve slightly altering the brightness or contrast of traffic sign images to test the model's robustness.
  - *Complex Input Perturbations:* This method introduces more complex perturbations to test robustness. Test the model's performance under extreme conditions, including varied lighting, outliers, and inputs that significantly deviate from the training data distribution or include substantial changes in input parameters. In the TSR example, this might include adding noise or occluding parts of traffic sign images to see how the model handles these challenges.
- **Adversarial Testing:** Evaluates the model's robustness against adversarial attacks.
  - *Without Physical Attacks:* Tests the model's robustness against digital adversarial inputs. This method tests the model's resistance to adversarial attacks that do not involve physical alterations. Generate adversarial examples using techniques such as gradient-based optimization or evolutionary algorithms to find input perturbations that result in incorrect predictions (e.g., generate perturbations from legitimate inputs to evaluate the model's robustness). Assess the effectiveness of defense mechanisms or adversarial training techniques in mitigating the impact of adversarial attacks on model performance. For a TSR classifier, this could involve generating adversarial examples digitally to test the model's resilience.
  - *With Physical Attacks:* Assesses the model's resilience against physical world adversarial scenarios. In the TSR example, this might involve placing stickers or marks on traffic signs to test if the model can still correctly classify them.
- **Neuron Coverage Analysis:** There are no variants envisaged for this method. This method analyzes the coverage of neurons during the model's operation. Assess and increase the coverage of activated neurons during inference to find model weaknesses. The coverage target should be 100% with the constraint to only use in-ODD data to increase the coverage. If a coverage of 100% is not achievable with in-ODD data or not

desired in a specific use case, a justification needs to be provided. For a TSR classifier, this would involve examining which neurons are activated by different traffic sign images to ensure the model uses its network comprehensively.

In the author's point of view, dropout and pruning do not conflict with neuron coverage analysis. Dropout and pruning aim to optimize the network architecture, while neuron coverage is a test method to check if the test cases are sufficient.

- **Explainability Testing:** Ensures that the decisions made by the ML model can be understood and interpreted.
  - *Simple Techniques:* Uses basic explainability methods to interpret model decisions. Evaluate the interpretability and explainability of the neural network model's predictions to facilitate understanding and trust in its decision-making process. Conduct thorough manual review or automated analysis of the data model, datasets, and associated metadata to understand the data and identify potential issues or deficiencies. Use interpretability techniques such as feature attribution methods, saliency maps, or attention mechanisms to visualize and explain the model's internal representations and decision boundaries. For a TSR classifier, this might involve using feature importance scores to show which parts of an image influenced the classification.
  - *Advanced Techniques:* Applies sophisticated explainability techniques to provide deeper insights into model behavior. Employ techniques such as SHAP values, LIME, and deep learning interpretability tools to analyze and explain complex model behaviors. Assess the robustness of the model's explanations under different scenarios and input variations to ensure they remain meaningful and accurate. In the TSR example, this could include using techniques like LIME[9] or SHAP[10] to provide detailed explanations of the model's predictions
- **Confidence Testing:** Assesses the model's ability to handle and communicate uncertainties in its predictions.
  - *Simple Input Data:* This method tests the model's confidence using simple input data. Evaluate the model's confidence level to ensure it aligns with the model's performance and predictions. Measure the calibration of the model's confidence scores with respect to prediction accuracy, particularly for probabilistic classifiers. This can be done using the Calibration Error metric (defined as Confidence Score − Observed Accuracy [2]). Evaluate the model's ability to provide meaningful confidence intervals or uncertainty bounds to indicate the reliability of its predictions under different conditions. For a TSR classifier, this might involve checking the confidence levels for standard traffic sign images
  - *Complex Input Data:* This method evaluates the model's confidence with more complex input data. Analyze the model's confidence estimates with complex and noisy input data to ensure robustness in varied conditions to assess the model's ability to maintain accurate confidence levels when dealing with inputs that deviate from the training data distribution. In the TSR use case, this could involve assessing confidence levels for images with multiple traffic signs or challenging environmental conditions

It is important to note that, in our approach, it is assumed that all test methods applied to the trained model are also applied to the deployed model, in addition to the specific test methods designed for the deployed model itself. This ensures comprehensive validation through back-to-back testing, confirming that the deployed model retains the functional properties and safety integrity established during the earlier phase. As such, the existing framework provides sufficient validation, and further techniques are not considered necessary at this stage.

E. MAPPING OF DESIRED PROPERTIES, METHODS FOR TRAINED ML MODEL AND RIGOR ASSIGNMENT

The desired properties for the trained ML model outlined in section V-C are mapped to specific test methods mentioned in section V-D to ensure they are effectively achieved and presented in Table 3. The methods are evaluated for their ability to support each desired property, with rigor levels assigned to indicate their effectiveness. Please note that the general rationale behind rigor assignment (i.e., R1 or R2) is outlined in section III-E. In this section, we detail how each method supports the desired properties.

F. MAPPING OF ASIL LEVEL AND TEST METHODS FOR TRAINED ML MODEL

The results from the mapping of ASIL levels and test methods for the trained ML model, and the recommendations are shown in Table 4. The process of assigning ASIL levels to different test methods for trained ML models is based on the method's rigor and its criticality in ensuring safety standards. This assignment, guided by expert judgment, carefully evaluates each method's capacity to meet the necessary confidence levels for safety. Methods categorized under ASIL A are recommended as highly rigorous (HR) for ensuring adequate assurance in less critical functions, while those deemed moderately rigorous (R) are suitable for more critical functions that demand heightened confidence. Conversely, methods classified under ASIL B are rated as highly recommended (HR) if they exhibit high rigor (R2), whereas those with moderate rigor (R1) are recommended (R).

[9]https://towardsdatascience.com/lime-explain-machine-learning-predictions-af8f18189bfe

[10]https://shap.readthedocs.io/en/latest/

TABLE 3. Mapping of desired properties and methods for trained ML model.

Method	Method Variant	Functional Correctness	Accuracy	Robustness	Uncertainty Handling	Explainability	Generalization
Requirements-Based Testing	No variants	R2					
Initial Performance Evaluation	Without Evaluations on Data Subsets		R1				
	With Evaluations on Data Subsets		R2				
Robustness Testing	Simple Input Perturbations			R1			
	Complex Input Perturbations			R2			
Adversarial Testing	Without Physical Attacks			R1		R1	R1
	With Physical Attacks			R2		R1	R1
Neuron Coverage Analysis	No variants					R1	R1
Explainability Testing	Simple Techniques					R1	
	Advanced Techniques					R2	
Confidence Testing	Simple Input Data		R1	R1	R1		R1
	Complex Input Data		R1	R1	R2		R1

TABLE 4. Test methods for the life cycle phase train ML model with ASIL mapping.

Method	Method Variant	ASIL A	ASIL B
Requirements-Based Testing	No variants	HR	HR
Initial Performance Evaluation	Without Evaluations on Data Subsets	HR	R
	With Evaluations on Data Subsets	R	HR
Robustness Testing	Simple Input Perturbations	HR	R
	Complex Input Perturbations	R	HR
Adversarial Testing	Without Physical Attacks	HR	HR
	With Physical Attacks	R	HR
Neuron Coverage Analysis	No variants	HR	HR
Explainability Testing	Simple Techniques	HR	R
	Advanced Techniques	R	HR
Confidence Testing	Simple Input Data	HR	R
	Complex Input Data	R	HR

VI. EVALUATION AND RESULTS FOR DEPLOYED ML MODEL

In this section, the systematic approach described in section III is applied to the third ML product life cycle phase, namely *deploy ML model*. Similar to ISO 26262, the objectives and inputs of the respective life cycle phase are defined. This is followed by a definition of a set of desired safety properties for the deploy ML model life cycle phase and a selection of test methods for this phase. The mapping of desired properties and test methods for the deployed ML model is subsequently outlined. Following the rigor assignment, a mapping of the ASIL level (A or B) and test methods for this life cycle phase is described. Thus, the presented approach provides a comprehensive framework for evaluating and ensuring the safety of deployed ML models in automotive systems. This framework includes:

- **Objectives:** Clearly defined goals to ensure the deployed model meets operational and safety requirements.
- **Inputs:** Necessary data and metadata, system requirements, and architectural specifications.
- **Desired Properties:** Identification of key safety properties such as functional correctness, accuracy, robustness, explainability, and uncertainty handling.
- **Test Methods:** Selection of appropriate methods to verify the deployed model.
- **Mapping of Properties to Methods:** Aligning desired properties with specific test methods to ensure a comprehensive evaluation.
- **Rigor Assignment:** Assigning rigor levels (R1 and R2) to test methods based on their effectiveness in achieving desired properties.
- **ASIL Mapping:** Recommendations for test methods based on ASIL A or ASIL B to guide their application in safety-critical contexts.

The purpose of the life cycle phase deploy ML model is to transfer the learned ML functionality to the target platform, enabling the integration of this functionality into the product and ensuring it can be deployed alongside other software units.

A. OBJECTIVES

- Ensure compatibility of the model with the target platform.
- Verify that the deployed model fulfills the allocated software requirements with the required ASIL.
- Ensure that the defined safety measures resulting from safety-oriented analyses are properly implemented.

B. INPUTS

- ML data model (in accordance with the prepare data life cycle phase).

- Trained model (in accordance with the train ML model life cycle phase).
- Software requirements (including ML specifics and target platform specifics).
- Software unit design specification.
- ML unit test specification of the trained ML model.

C. DESIRED PROPERTIES FOR THE LIFE CYCLE PHASE DEPLOY ML MODEL

In the following, the safety desired properties for the identified third life cycle phase, namely deploy ML model, are discussed. Each property is introduced with a description and a simple example. The examples, based on a TSR classifier use case, provide *representative scenarios* demonstrating how these properties apply in practice.

- **Compatibility:** The goal here is to ensure that the deployed model is compatible with the target environment and retains the learned functionality. Compatibility refers to the need to ensure that the deployed model works seamlessly within the target hardware, software, and operational environment, maintaining the functional integrity of the trained ML model. This involves verifying that the deployed model meets all relevant system and software requirements.
  Example: For a TSR classifier, compatibility would mean that the model, when deployed from a development environment to an in-vehicle platform, continues to accurately recognize and classify traffic signs without any degradation in performance.
- **Portability:** The aim here is to ensure that the model can be efficiently migrated or adapted to different target platforms. Portability involves the ability to transfer the trained ML model to various hardware and software environments without loss of functionality or performance. This includes considerations of different hardware configurations, operating systems, and software dependencies.
  Example: For a TSR classifier, portability would involve ensuring that the model can be moved from a high-performance server environment to an embedded system in a vehicle while maintaining its accuracy and efficiency. This includes evaluating metrics such as speed, memory usage, and resource consumption in the new environment.

D. TEST METHODS FOR THE LIFE CYCLE PHASE DEPLOY ML MODEL

This section outlines the test methods selected for the deploy ML model life cycle phase. Each method is carefully chosen to ensure the model meets compatibility and performance requirements on the target platform. The methods are categorized and detailed to provide a comprehensive framework for evaluating the deployed model. The selection of these methods corresponds with guidelines provided in ISO 29119, Chapter 8 [6], which focuses on black-box testing of AI-based systems, specifically Section 8.2, addressing back-to-back testing. This technique is critical for ensuring that the deployed model's behavior remains consistent with the original model's behavior, particularly after deployment on the target platform. Additionally, the methods align with ISO/IEC TR 5469 [3], Chapter 9, which covers verification and validation techniques. Section 9.4 emphasizes the importance of both virtual and physical testing, ensuring that the deployed model not only meets functional requirements but also performs reliably in real-world environments.

- **Compatibility Testing:** Ensures that the deployed ML model is compatible with the target platform's hardware, software, operating system, and other dependencies.
  - *Without AI-in-the-loop:* Verify that the deployed model is compatible with the NN model by testing the desired properties of the NN model on the deployed model and analyzing the comparison for deviations. Perform extensive back-to-back tests with the NN model by applying all applicable test methods from the NN life cycle phase according to the ASIL recommendations and analyze the results for unexpected deviations. Use a simulated environment or the target hardware to execute the tests on the deployed model. Verifying that the deployed model is compatible with the target platform's hardware, software, operating system, and other dependencies without any compatibility issues or conflicts. Testing the model's compatibility across different relevant versions, configurations, and environments to ensure seamless integration and interoperability. Measuring the consistency of the model's prediction across different relevant platforms, e.g., comparing the deployed model's predictions with the NN model using back-to-back testing
  - *With AI-in-the-loop:* Verify that the deployed model is compatible with the NN model by testing the desired properties of the NN model on the deployed model and analyzing the comparison for deviations. Perform extensive back-to-back tests with the NN model by applying all applicable test methods from the NN life cycle phase according to the ASIL recommendations and analyze the results for unexpected deviations. Use an "AI in the loop" setup to perform the tests on the deployed model. Testing real-time compatibility and integration with other system components during active ML operations. Evaluating performance metrics under actual operating conditions to identify any issues that may arise during real-world deployment.
- **Portability Testing with Hardware-Related Performance Evaluation:** Assesses the performance of the deployed ML model to ensure it meets the required benchmarks on the target hardware.
  - *With Average Workload:* Evaluates the model's performance under standard operating conditions.

TABLE 5. Mapping of desired properties and methods for the deployed ML model.

Method	Method Variant	Compatibility	Portability
Compatibility Testing	Without AI-in-the-loop	R1	-
	With AI-in-the-loop	R2	-
Portability Testing	With average workload	-	R1
	With high workload	-	R2

TABLE 6. Test methods for the life cycle phase deploy ML model with ASIL mapping.

Method	Method Variant	ASIL A	ASIL B
Compatibility Testing	Without AI-in-the-loop	HR	R
	With AI-in-the-loop	R	HR
Portability Testing	With average workload	HR	R
	With high workload	R	HR

Evaluate the fulfillment of nonfunctional requirements like speed, efficiency, resource utilization, and scalability as part of the system or software requirements. Perform the evaluation for relevant target platforms. Measuring key performance indicators such as inference latency, throughput, memory usage, and CPU/GPU utilization under average workloads and conditions. Comparing the performance of the deployed model against predefined benchmarks and performance targets as defined in the system requirements to assess its effectiveness and suitability for deployment.

– *With High Workload:* Tests the model's performance under heavy load conditions to ensure robustness and reliability. Evaluate the fulfillment of nonfunctional requirements like speed, efficiency, resource utilization, and scalability as part of the system or software requirements and evaluate the model behavior under high workload. Measuring the impact of high workload conditions on key performance indicators such as inference latency, throughput, memory usage, and CPU/GPU utilization. Ensuring the model remains stable and performs efficiently even under maximum operational stress. Perform the evaluation for all target platforms.

Thus, the methods provided above ensure that the deployed ML model is thoroughly evaluated for compatibility and performance, enabling it can be effectively integrated and deployed on the target platform.

E. MAPPING OF DESIRED PROPERTIES AND METHODS FOR THE DEPLOYED ML MODEL AND RIGOR ASSIGNMENT

The desired properties for the deployed ML model outlined in section VI-C are mapped to specific test methods mentioned in section VI-D to ensure they are effectively achieved and presented in Table 5. The methods are evaluated for their ability to support each desired property, with rigor levels assigned to indicate their effectiveness. Please note that the general rationale behind rigor assignment (i.e., R1 or R2) is outlined in section III-E.

Table 5 and the descriptions above illustrate how each method supports key properties such as compatibility and portability. The rigor levels (R1 and R2) indicate the method's effectiveness, with R2 representing a higher level of rigor.

F. MAPPING OF ASIL LEVEL AND TEST METHODS FOR THE DEPLOYED ML MODEL

The results from the mapping of ASIL levels and test methods for the deployed ML model, and the recommendations, are shown in Table 6. The process of assigning ASIL levels to different test methods for deployed ML models is based on the method's rigor and criticality in ensuring safety standards. This assignment, guided by expert judgment, carefully evaluates each method's capacity to meet the necessary confidence levels for safety. Methods categorized under ASIL A are recommended as highly rigorous (HR) for ensuring adequate assurance in less critical functions, while those deemed moderately rigorous (R) are suitable for more critical functions that demand heightened confidence. Conversely, methods classified under ASIL B are rated as highly recommended (HR) if they exhibit high rigor, whereas those with moderate rigor are recommended (R).

VII. DISCUSSION

This section provides a discussion of the results obtained from the systematic approach applied to various life cycle phases of ML models in automotive systems, as detailed in the previous sections. The phases include prepare data, the trained ML model, and the deployed ML model. In this work, each phase's objectives, inputs, test methods, and desired properties were rigorously defined and evaluated to ensure the safety and reliability of ML models.

A. PREPARE DATA

The evaluation of the ML data model highlighted the critical importance of ensuring data completeness, consistency, correctness, representativeness, and independence. The systematic approach can be used to effectively identify and mitigate data-related issues, providing a robust foundation

for subsequent phases. By ensuring high-quality data, the risk of biases and inaccuracies can be significantly reduced, leading to more reliable and accurate ML models. This phase's results impact the entire ML life cycle by establishing a solid data foundation essential for training effective and safe models.

B. TRAIN ML MODEL

For the trained ML model, the evaluation framework emphasized desired safety properties such as functional correctness, accuracy, robustness, explainability, and uncertainty handling. Employing the systematic approach would enable the trained models to meet the specified software requirements and perform reliably across various conditions. The rigorous testing, including robustness and adversarial testing, can be used to ensure that the models handle diverse inputs and perturbations, thus enhancing their reliability in real-world scenarios.

C. DEPLOY ML MODEL

The deployed ML model's evaluation concentrates on safety desired properties such as compatibility and portability across different hardware platforms and environments. The testing methods can be used to verify that the deployed model maintains the learned functionality and performance when deployed on the target platform. Further, employing these methods would help us ensure that the deployed model met the required ASIL levels and could be seamlessly integrated into automotive systems. These results impact the final deployment phase, ensuring that the ML models can be reliably transferred and operated in their intended environments without loss of functionality or performance.

D. CONSIDERATIONS FOR EXTENDING THE APPROACH TO ASIL C AND D LEVELS

While this paper primarily focuses on enhancing ISO 26262 for ML-specific life cycle phases at ASIL A and B levels, it is crucial to consider the implications and necessary adaptations for higher safety integrity levels, such as ASIL C and D. These levels represent a more stringent requirement for safety-critical systems, where the consequences of failure are more severe, potentially leading to significant harm or fatality. Extending the proposed approach to these levels requires additional rigor and verification steps to ensure the robustness and reliability of ML models in more critical applications. In the following, some of the challenges foreseen and proposed adaptations in extending the approach to ASIL C and D, at this juncture, are outlined. Please note that, at this juncture, given the lack of field data and real-world experience at ASIL C and D levels, this paper presents a foundational step for future work. Key aspects such as redundancy and fail-operational systems, which are crucial at these higher levels, have yet to be fully evaluated but will be prioritized in ongoing research.

1) CHALLENGES AT ASIL C AND D LEVELS

The primary challenges in extending the approach to ASIL C and D include:

- Increased Rigor in Verification and Validation: ASIL C and D demand higher levels of confidence in the safety and reliability of the system. This necessitates more rigorous verification and validation methods, including exhaustive testing under a wider range of conditions and scenarios. Techniques such as formal methods, fault injection testing, and more comprehensive safety case documentation would be required.
- Enhanced Safety Measures: At these higher ASIL levels, the safety measures must be more robust. This could include additional layers of redundancy, fail-safes, and real-time monitoring systems to detect and mitigate potential failures. The ML models would need to be designed with these considerations in mind, ensuring that they can operate safely even in the presence of unexpected inputs or conditions.
- Stricter Requirements for Data Handling: The data used for training and testing ML models must meet stricter quality and traceability requirements at ASIL C and D. This includes ensuring that data is not only complete and consistent but also fully traceable, with documented evidence of its provenance and processing history. Any biases or anomalies in the data could have more severe consequences at these levels, so rigorous data governance practices would be essential.
- Increased Focus on Explainability and Interpretability: For ASIL C and D applications, it is critical that the decision-making processes of ML models are not only accurate but also explainable. This is important for gaining regulatory approval and for ensuring that the system's behavior can be understood and trusted by human operators, especially in safety-critical scenarios.

2) PROPOSED ADAPTATIONS

To extend the systematic approach to ASIL C and D, the following adaptations are proposed:

- Formal Verification Techniques: Incorporating formal verification techniques, such as model checking and theorem proving, could provide the additional assurance needed for ASIL C and D. These techniques can mathematically prove the correctness of certain aspects of the system, which is particularly valuable at higher safety integrity levels.
- Integration of Advanced Safety Mechanisms: The inclusion of advanced safety mechanisms, such as runtime verification, fault-tolerant architectures, and real-time anomaly detection, would be necessary to meet the stringent requirements of ASIL C and D. These mechanisms would help ensure that the system can continue to operate safely even in the presence of faults or unexpected conditions.

- Stricter Data Governance and Management: Extending the approach to higher ASIL levels would require stricter data governance practices, including more detailed documentation and traceability of data sources, processing steps, and validation results. This would help ensure that the data used to train and test ML models is of the highest possible quality and reliability.

E. IMPLICATIONS

The proposed approach significantly enhances the development and deployment of ML models in automotive systems. Rigorous testing and evaluation at each life cycle phase improve overall safety and reliability, identifying and mitigating potential issues early in development. The approach supports standardization of AI testing practices within the automotive industry, aligning with ISO 26262, ensuring compliance, and promoting consistency in safety assessments. It also provides clear guidelines for AI developers and safety engineers, fostering a culture of safety and reliability, building trust among stakeholders, and enhancing operational safety and efficiency in automotive systems.

F. THREATS TO VALIDITY AND LIMITATIONS

This section discusses potential threats or limitations of the proposed approach, focusing on areas such as generalizability, quantitative measurement, flexibility, and practical implementation.

1) GENERALIZABILITY

The TSR classifier is used solely as an illustrative example to explain the approach, not as a comprehensive case study. While helpful for didactical clarity, it does not account for the range of conditions in real-world applications, which limits generalizability. Additionally, empirical validation through structured case studies, as outlined by [36], has not been conducted. The absence of empirical evaluation presents a limitation in assessing the practical applicability and scalability of the approach. Future work should validate the approach using diverse real-world scenarios and well-designed case studies to provide stronger evidence for its effectiveness and generalizability.

2) QUANTITATIVE MEASUREMENT CHALLENGES

A challenge arises from the difficulty in obtaining structured quantitative metrics to evaluate aspects such as performance, efficiency, and scalability. While qualitative insights are valuable, developing concrete metrics for quantitative analysis will enhance the robustness of the approach in future applications.

3) FLEXIBILITY AND COVERAGE LIMITATIONS

In this paper, the approach has only been applied to the level of examples for the TSR. The approach's flexibility needs to be validated by e.g., applying the concept to several practical use cases with different conditions like model size or model architecture. Furthermore, the approach primarily addresses testing activities, aligning with the right side of the V-model in system development processes. It does not encompass systematic development methods required for the left side of the V-model for ML models. This includes the absence of ASIL-suitable methods for requirements engineering, architectural design, and other preliminary development phases. Moreover, non-AI-specific data aspects, which are highly relevant in safety-critical systems, are not covered by our approach.

4) PRACTICAL IMPLEMENTATION CHALLENGES

Implementing the proposed approach in practical environments may require significant resources, including computational power and expertise. This could present challenges, especially for smaller organizations. Additionally, the need for continuous updates to align with evolving ML standards poses long-term sustainability concerns. Future research should focus on making the approach more resource-efficient and adaptable to various organizational capacities. Thus, the limitations identified, including those related to generalizability, quantitative measurement, flexibility, and practical implementation, highlight areas for further research. Addressing these issues will enhance the overall applicability and robustness of the proposed approach across different domains and use cases.

G. FUTURE DIRECTIONS

Future work should focus on integrating the approach into the ISO 26262 framework and certification processes, developing clear certification protocols, and creating automated tools to streamline testing and evaluation. For instance, collaboration with the ISO 26262 working committee could help to integrate these ML-specific life cycle phases and testing methods into the standard. This collaboration aims to standardize the approach and promote widespread adoption, ultimately contributing to safer and more reliable automotive systems. Additionally, incorporating Continuous Integration/Continuous Deployment (CI/CD) pipelines will further streamline the testing and deployment process, ensuring that updates and new models are seamlessly integrated while maintaining rigorous safety standards. Last, but not the least, applying the framework to diverse ML models and automotive use cases will help validate its effectiveness and adaptability.

VIII. CONCLUSION

The novel, systematic approach presented in this paper identifies and addresses the gaps in the widely adopted standard for automotive functional safety, namely, ISO 26262 [1]. While standards such as ISO PAS 8800 [2] provide an approach for developing AI systems, it does not provide a mapping concept for ASIL classification of ML systems, revealing significant gaps. The approach elaborated in this paper outlines ML-specific life cycle phases, desired safety properties, and testing methods, supporting effectiveness

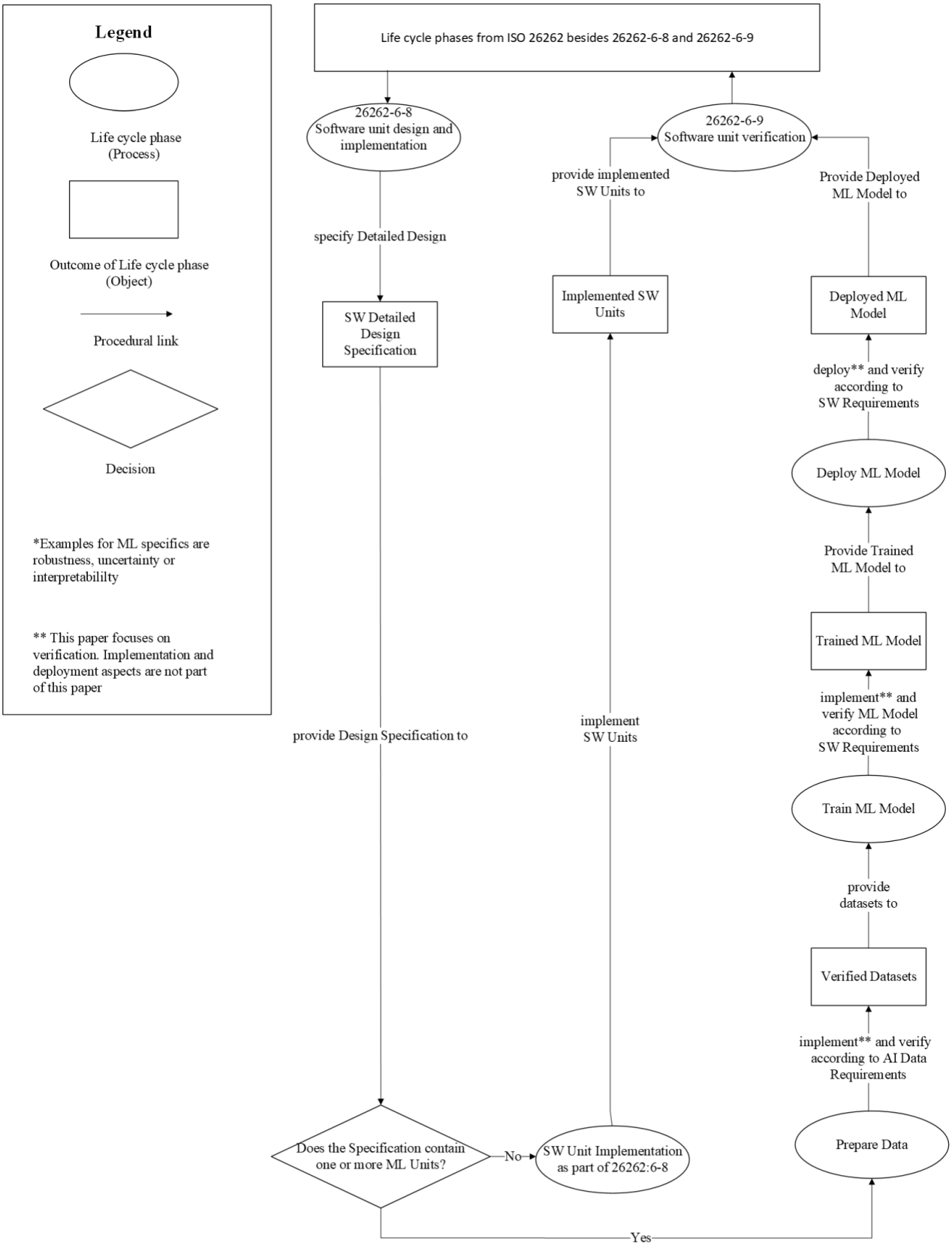

FIGURE 6. Detailed interfaces of ML life cycle phases introduced in this paper with ISO 26262-Part 6.

in ensuring the safety and reliability of ML models in automotive systems. The structured, rigorous framework provides clear guidelines for testing and evaluation, addressing critical aspects and prioritizing testing efforts based on function criticality. A novel mapping concept for these testing methods to ASIL A/B level classification is also outlined. Moreover, this paper ensures that each desired safety property is covered by at least one test method across all three phases—data model, trained model, and deployed model. While additional test methods can be introduced in future research, our focus was to prioritize comprehensive coverage of key safety properties. Thus, this comprehensive approach offers a valuable tool for developers and safety engineers, facilitating the safe and reliable integration of ML models into vehicles and enhancing the safety and performance of automotive systems. We aim to collaborate with the ISO 26262 working committee to incorporate these enhancements into future iterations of the standard.

APPENDIX A

Fig. 6 provides more detail on how the introduced life cycle phases are integrated into the traditional V-Model and aligned with the life cycle phases of ISO 26262 Part 6, as an extension to Fig. 4 in section III. In order to support the readability of Fig. 6, not all life cycle phases of ISO 26262 are displayed. From the two life cycle phases of ISO 26262 that are part of this figure, only some of the defined outcomes are displayed. Only those outcomes are displayed, which are considered the most relevant interfaces to the introduced life cycle phases. This figure assumes, that the ML model is part of the detailed design and will be tested with other SW Units according to ISO 26262-6-9. As seen in Fig. 6, the Software (SW) Detailed Design Specification serves as the branching point, where it is determined whether the specification contains ML components or if the traditional SW Unit Design process is followed. Please note that, while Fig. 6 is conceptual and focused on readability, we acknowledge that formal system modeling languages such as SysML ISO/IEC 19514 [37] or OPM ISO 19450 [38] may provide a more structured representation of the processes. The choice of a simplified diagram in this case was made to ensure that the content is easily understandable for a broad audience in the automotive safety domain.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to the anonymous reviewers for their valuable suggestions, which have significantly enhanced the quality of this paper. Our thanks also go to our colleagues from CARIAD and innotec GmbH for their insightful feedback during numerous discussions, which have strengthened the technical soundness of the paper. Additionally, we extend our appreciation to the experts from TRUSTIFAI[11] a joint venture of TÜV AUSTRIA and SCCH (Software Competence Center Hagenberg), for their invaluable participation in the expert judgment process, contributing to the evaluation of this framework.

[11] https://www.trustifai.at/

REFERENCES

[1] *Road Vehicles—Functional Safety*, Standard ISO 26262, 2018.
[2] *Road Vehicles—Safety and AI*, Standard SO Std. ISO/DPAS 8800, 2023.
[3] *Artificial Intelligence—Functional Safety and AI Systems*, Standard ISO/IEC JTC1/SC2, 2024.
[4] *Functional Safety of Electrical/electronic/programmable Electronic Safetyrelated Systems*, Standard IEC 61508-1:2010, 2010.
[5] *Road Vehicles—Functional Safety—Safety and AI for Automated Driving Systems*, Standard ISO Std. ISO 21 448, 2022.
[6] *Software and Systems Engineering—Software Testing—Part 11: Guidelines on the Testing of AI-based Systems*, Standard ISO/IEC TR 29119-11:2020, 2020.
[7] *Information Technology—Artificial Intelligence—AI System Life Cycle Processes*, Standard ISO/IEC JTC 1/SC 42, 2023.
[8] *Systems and Software Engineering—System Life Cycle Processes*, Standard ISO/IEC JTC 1/SC 7, 2023.
[9] *Systems and Software Engineering—Software Life Cycle Processes*, Standard ISO/IEC JTC 1/SC 7, 2017.
[10] *Information Technology—Artificial Intelligence—Artificial Intelligence Concepts and Terminology*, Standard ISO/IEC JTC 1/SC 42, 2022.
[11] *Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)*, Standard ISO/IEC JTC 1/SC 42, 2022.
[12] E. Gracic, F. Svensson, J. Ehrich, O. Beck, and M. Jansen, "Concept for safety-related development of deep neural networks in the automotive," in *Proc. 4th Int. Conf. Multimedia Comput., Netw. Appl. (MCNA)*, 2020, pp. 10–15.
[13] (2022). *ASAM OpenSCENARIO*. [Online]. Available: https://www.asam.net/standards/detail/openscenario/
[14] D. Dori, "Model-based standards authoring: ISO 15288 as a case in point," *Syst. Eng.*, vol. 27, no. 2, pp. 302–314, Mar. 2024, doi: 10.1002/SYS.21721.
[15] (2024). *KI Absicherung Project: SAFE AI for Automated Driving*. Accessed: Jul. 2, 2024. [Online]. Available: https://www.ki-absicherung-projekt.de/en/
[16] *Road Vehicles—Safety and Cybersecurity for Automated Driving Systems—Design, Verification and Validation*, Standard ISO/TR 4804:2020, 2010, p. 2020.
[17] J. Henriksson, M. Borg, and C. Englund, "Automotive safety and machine learning: Initial results from a study on how to adapt the ISO 26262 safety standard," in *Proc. IEEE/ACM 1st Int. Workshop Softw. Eng. AI Auto. Syst. (SEFAIAS)*, May 2018, pp. 47–49.
[18] R. Salay and K. Czarnecki, "Using machine learning safely in automotive software: An assessment and adaption of software process requirements in ISO 26262," 2018, *arXiv:1808.01614*.
[19] S. Mohseni, M. Pitale, V. Singh, and Z. Wang, "Practical solutions for machine learning safety in autonomous vehicles," 2019, *arXiv:1912.09630*.
[20] J. Serna, S. Diemert, L. Millet, R. Debouk, R. S, and J. Joyce, "Bridging the gap between ISO 26262 and machine learning: A survey of techniques for developing confidence in machine learning systems," *SAE Int. J. Adv. Current Practices Mobility*, vol. 2, no. 3, pp. 1538–1550, Apr. 2020, doi: 10.4271/2020-01-0738.
[21] J. Thomas, "Integrating machine learning and AI in automotive safety: Enhancing ISO 26262 compliance," *Int. J. Innov. Sci. Res. Technol.*, vol. 9, no. 1, pp. 1–24, 2024.
[22] F. Pourdanesh, T. Q. Dinh, F. Tagliabo, and P. Whiffin, "Failure safety analysis of artificial intelligence systems for Smart/Autonomous vehicles," in *Proc. 24th Int. Conf. Mechatronics Technol. (ICMT)*, Dec. 2021, pp. 1–6.
[23] S. Abrecht, A. Hirsch, S. Raafatnia, and M. Woehrle, "Deep learning safety concerns in automated driving perception," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 1, pp. 1–12, Jun. 2024.
[24] K. Radlak, M. Szczepankiewicz, T. Jones, and P. Serwa, "Organization of machine learning based product development as per ISO 26262 and ISO/PAS 21448," 2019, *arXiv:1910.05112*.
[25] J. Perez-Cerrolaza, J. Abella, M. Borg, C. Donzella, J. Cerquides, F. J. Cazorla, C. Englund, M. Tauber, G. Nikolakopoulos, and J. L. Flores, "Artificial intelligence for safety-critical systems in industrial and transportation domains: A survey," *ACM Comput. Surveys*, vol. 56, no. 7, pp. 1–40, Apr. 2024.
[26] K. Rana and N. Khatri, "Automotive intelligence: Unleashing the potential of AI beyond advance driver assisting system, a comprehensive review," *Comput. Electr. Eng.*, vol. 117, Jul. 2024, Art. no. 109237.

[27] S. Diemert, L. Millet, J. Groves, and J. Joyce, "Safety integrity levels for artificial intelligence," in *Proc. Comput. Saf., Rel., Security. SAFECOMP Workshops*, 2023, pp. 397–409.
[28] G. Stettinger, P. Weissensteiner, and S. Khastgir, "Trustworthiness assurance assessment for high-risk AI-based systems," *IEEE Access*, vol. 12, pp. 22718–22745, 2024.
[29] A. V. S. Neto, J. B. Camargo, J. R. Almeida, and P. S. Cugnasca, "Safety assurance of artificial intelligence-based systems: A systematic literature review on the state of the art and guidelines for future work," *IEEE Access*, vol. 10, pp. 130733–130770, 2022.
[30] S. Dey and S.-W. Lee, "Multilayered review of safety approaches for machine learning-based systems in the days of AI," *J. Syst. Softw.*, vol. 176, Jun. 2021, Art. no. 110941.
[31] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.
[32] (2023). *Automatic SPICE Process Assessment/Reference Model*. [Online]. Available: https://vda-qmc.de/wp-content/uploads/2023/12/Automotive-SPICE-PAM-v40.pdf
[33] M. R. Steenbergen and G. Marks, "Evaluating expert judgments," *Eur. J. Political Res.*, vol. 46, no. 3, pp. 347–366, May 2007.
[34] L. M. Restrepo-Tamayo and G. P. Gasca-Hurtado, "Gamified focus group for empirical research in software engineering: A case study," in *Communications in Computer and Information Science*. Cham, Switzerland: Springer, 2023, pp. 59–71.
[35] *Failure Mode and Effects Analysis (FMEA) Handbook*, 1st ed. Berlin, Germany: AIAG and VDA, Jun. 2019. [Online]. Available: https://webshop.vda.de/QMC/en/aiag-vda-fmea-handbook_eng
[36] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. E. Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *IEEE Trans. Softw. Eng.*, vol. 28, no. 8, pp. 721–734, Aug. 2002.
[37] *Information Technology—Object Management Group Systems Modeling Language (OMG SysML)*, Standard ISO/IEC JTC1, 2017.
[38] *Automation Systems and Integration—Object-Process Methodology (OPM)*, Standard ISO 19450:2024, 2024.

PADMA IYENGHAR received the B.E. degree in computer science in India, in 2003, the M.S. degree in computer science from the Technische Universität Hamburg (TUHH), in 2006, and the Ph.D. degree in computer science from the University of Osnabrück (UOS), Germany, in 2012.

Until October 2020, she was a Postdoctoral Researcher with UOS, where she led and contributed to various publicly funded research and innovation projects in collaboration with industrial partners. Since 2020, she has been a Senior Functional Safety and Cybersecurity Consultant with innotec GmbH-TÜV Austria Group and concurrently a Development Professor with the University of Applied Sciences, Osnabrück, Germany, balancing her professional commitments across teaching, research, and industry engagements. She has over 14 years of research and development experience in model-based software development and embedded software engineering and more than eight years of teaching experience in various computer science subjects. She has supervised numerous theses and published in reputed conferences and journals. Additionally, she is active in academic and professional communities, serving as a guest editor, organizing special sessions at conferences, and holding editorial and reviewer roles for various journals. At innotec GmbH, she has been involved in all phases of the functional safety life cycle, working across industries, such as machinery, automation, railway, and automotive. She leads projects pertaining to AI and functional safety and AI and cybersecurity and is an active member of DKE/AK committees on functional safety and AI. Her academic achievements include the Best Outgoing Student Award and the University Medal for Excellence in Academics during her undergraduate studies.

EMIL GRACIC received the Ph.D. degree in computer science from the Department for Computer Architecture and System Programming, University of Kassel, Germany.

In parallel, he worked for six years as a Research Assistant on projects for the industry partner, the company HIMA, where he was involved in the chip design of HiCore2 SoC. From July 2018 to January 2020, he was with the ADAS Department, Hella Aglaia, Berlin, as a Functional Safety Manager, involved in investigating and implementing ASIL B concepts in the context of various perception software modules. Since February 2020, he has been with CARIAD SE as an AI Safety Expert focusing on processes, methods, and tools for certifiable AI modules. He is the author of many scientific articles related to functional safety and FPGAs. His last publication about an adequate and reliable process for ML development in automotive still resonates within different industries as a very plausible process model. His research interests include the implementation and evaluation of on-chip redundancy on the FPGA platform.

GREGOR PAWELKE received the degree in industrial engineering. He focused on the development of a process model for ML development in the automotive sector as part of his master's thesis. Until 2021, he was a Quality Assurance Expert with Hella Aglaia and supported teams in achieving their quality goals and ASPICE compliance. In parallel, he developed concepts to ensure quality in development projects with ML functionality by adapting and extending known concepts and frameworks with ML specifics. Since 2021, he has been with CARIAD on processes, methods and tools for the development of ML functionalities in safety-critical projects.