Organization of ML-based product development as per ISO 26262

Krystian Radlak  
exida  
Warsaw, Poland  
kradlak@exida.com

Michał Szczechankiewicz  
exida  
Warsaw, Poland  
msz@exida.com

Piotr Serwa  
exida  
Warsaw, Poland  
pserwa@exida.com

Tim Jones  
exida  
Erlangen, Germany  
tim.jones@exida.com

ABSTRACT
Machine learning (ML) applications generate a continuous stream of success stories from various domains. ML enables many novel applications, also in a safety-related context. With the advent of Autonomous Driving, ML gets used in automotive domain. In such a context, ML-based systems are safety-related. In the automotive industry, the applicable functional safety standard is ISO 26262, which does not cover specific aspects of ML. In a safety-related ML project, all ISO 26262 work products are typically necessary and have to be delivered. However, specific aspects of ML (like data set requirements, special analyses for ML) must be addressed within some work products. In this paper, we propose how the key technical aspects and supporting processes related to development of ML-based systems can be organized according to ISO 26262 phases, sub-phases and work-products.

KEYWORDS
autonomous driving, automotive software, machine learning, dependability, functional safety, ISO 26262, software engineering

1 INTRODUCTION
Along with utilizing machine learning (ML) algorithms the quality of several non-safety related products has improved in the past years. Capabilities of those algorithms to learn and work with incomplete knowledge and their generalization capabilities make them highly desirable solutions for complex problems that currently may not even have well-known analytical solution. This has motivated the introduction of ML techniques/components into products from many industry domains including automotive systems.

In automotive industry ML algorithms are the core of Autonomous Driving as they allow to interpret and understand the surrounding environment and make driving-related decisions. Such systems are safety-related, since their malfunction may cause death or injury. Therefore, deployment of ML algorithm into such applications must follow a rigorous development cycle according to state-of-art and according to applicable safety standards. However, the use of ML algorithms is not sufficiently covered by existing automotive safety standards such as ISO 26262 [11] or Safety of the Intended Function (SOTIF) [12]. Even more, IEC 61508 [10] standard explicitly recommends not to use artificial intelligence algorithms. For this reason, existing safety standards and their verification/validation techniques cannot be directly applied for ML algorithms, as they do not properly address the special characteristics of ML-based components such as non-determinism, non-transparency.

Automotive AD/ML projects are safety-related, thus ISO 26262 needs to be followed, both at the process level (e.g. change management, configuration management) and technical level (e.g. how to build and verify safe software or hardware). ISO 26262 defines a well-known structure/framework for safety-related automotive projects. All ISO 26262 work products are to be provided, but some of them need to be extended to cover specific aspects of ML/AD (e.g. datasets requirements, special analyses for ML, SOTIF/SAE at item level). Moreover, also integration of various software components is needed (e.g. standard C-implemented components and ML code).

At this moment, academic approaches do not provide a complete solution to the lack of understandability of ML algorithms, including deep neural networks [9]. Moreover, academic work typically focuses on performance issues and ignores safety aspects in work (e.g. such us potential failure modes and safety measures that could mitigate potential faults of algorithms and data). Additionally, the whole life cycle of the ML algorithm development (e.g. including data collection, labeling process, algorithms design and development) is not taken into account. For example, together with one of the most popular dataset KITTI 3D object detection [7], the authors provide a benchmark that ranks object detection algorithms according to the average precision metric. The researchers that published their ML-based solutions focused only on this single evaluation metric, whereas some of current limitations of the published algorithms might be caused by insufficient quality of the objects’ labeling, e.g. missing objects, annotations, improper training and testing datasets data distribution or insufficient parameters optimization. Complete analysis of the whole life cycle of a concrete ML component from item-level hazard analysis to deployment of binary code on a hardware platform exceeds resources of any academic research group. For instance, in [31] the authors to this date identified 1470 individual hazards that could affect computer
vision algorithms. Coverage of these hazards by the test dataset, definition of safety mechanisms that will mitigate these hazards and integration of these mechanisms into the developed algorithm is required in order to eliminate long-term risks or mitigate them to the acceptable level. Such activities are resource-consuming and potential exhaustive tackling of them is even beyond the capabilities of existing manufacturers and requires a consolidated cooperation between automotive industry and academia.

The main goal of this paper is to propose how development process of ML-based components can be oriented towards well-known ISO 26262 practices that have been utilized for years, instead of creating a completely new standard not yet exposed to the automotive industry. In this work, we present how ML based components can be integrated into currently defined work products in ISO 26262 starting from item definition and finishing on safety validation. To simplify the general understanding of our proposal, we present a simplified example of a road line detection.

This document shall provide guidance by addressing technical key aspects related to the development of ML-based systems. A manufacturer will very likely have to be able to present and argue these essential aspects during an assessment and provide evidence that their system is safe. However, this document is not intended to provide detailed guidance on all aspects of a safety lifecycle and therefore by nature is not entirely complete. It only points out main objectives.

The paper is structured as follows: section II describes the existing automotive standards and existing work in which authors discuss how to ensure safety and use ML in automotive industry. Section III presents our proposal on how ISO 26262 could be applied to an autonomous vehicle that utilizes a machine learning algorithm to provide a certain safety-related functionality. It is also pointed out which work products defined in ISO 26262 require a special attention when a vehicle is extended with a new item that uses ML. Finally, the conclusions are given in Section IV.

2 RELATED WORK AND EXISTING STANDARDS

The main purpose of ISO/PAS 21448 (SOTIF) is to cover a foreseeable misuse of a system by a driver, as well as the system/technological shortcomings of a product. Malfunctions of E/E components are addressed by ISO 26262 standard. The risk coming from a deliberate impact is covered by security standard ISO/SAE 21434. Organizing a development process as per ISO/PAS 21448, ISO 26262 and ISO/SAE 21434 and utilizing their well-established global state-of-art practices is a key to achieve a safe/secure/dependable ML-based autonomous system.

The ISO has recently created a new technical subcommittee (ISO/JTC 1 SC 42) that is to operate in the area of artificial intelligence. Scope of work covers fundamental standards as well as issues related to safety and trustworthiness [19].

Salay et al. analyze ISO 26262 from an ML perspective and they discuss five topics that should be addressed in development of ML based components such as: new types of hazards specific for ML and new types of faults and failure modes, usage of incomplete training datasets, level on which ML algorithms should be used, and which software techniques should be required in their verification [21, 23]. They also analyzed software verification methods recommended by ISO 26262 and concluded that most of verification techniques can by directly applied or adopted for ML algorithms.

Cheng et al. introduced a novel tool called the NN-Dependability Kit to support safe design and development of deep neural networks (DNN) for autonomous driving systems [4]. This tool offers several state-of-the-art techniques that could improve safety engineering of DNN including dependability metrics, techniques for ensuring that the generalization does not lead to undesired behaviors and runtime monitoring methods.

Henriksson et al. proposed that the ML component can be realized as a software unit on the software level phase and discuss the most critical gaps between ISO 26262 and ML development [8]. Additionally, they propose three adaptations related to ML training, model sensitivity, and test case design that are crucial to develop ML based components according to the ISO 26262.

Molina et al. proposed to implement an independent module - the Autonomous Vehicle Control (AVC)- that is going to both interact with the vehicle’s systems and create a protection layer that is independent of the way the vehicle’s system was developed. So, the AVC could be used with any autonomous vehicle system and could be tested individually [17].

Koopman et al. proposed to introduce a novel Standard for Safety for the Evaluation of Autonomous Products (UL 4600) that is intended to cover autonomous driving and eventually other related domains [14].

Spanfelner et al. [25] highlighted that perception functionality for autonomous driving may not be completely specifiable and human categories (e.g., pedestrians) can only partially be specified using rules (e.g., necessary and sufficient conditions) and also need examples. This topic has also been addressed in [22], where authors proposed to use partial specifications instead of complete specifications, which are difficult or impossible to specify precisely.

Another interesting overview of problems related to application of ML methods in autonomous driving was presented in the white paper called Safety First for Automated Driving (SaFAD) [26]. This technical report was released by 11 automotive companies and key technology providers, the authors presented the summary how to develop and validate a safe automated driving system. They also proposed to divide the development of ML components into four steps define, specify, develop & evaluate and deploy & monitor.

3 PROPOSED ORGANIZATION OF ML-BASED PRODUCT DEVELOPMENT AS PER ISO 26262

Providing a dedicated standard that is sufficiently detailed in addressing crucial aspects of design and verification for application of ML algorithms in the automotive domain would require a lot of work and time for its creation, careful review process and reaching an agreement across the whole safety experts community. In this work, instead of delivering a stand-alone standard, we propose a comprehensive solution that is aligned to the well-know and widely applied ISO 26262.

We recommend this approach due to following reasons:
ISO 26262 is a well established standard with well defined terminology and has been used by automotive industry and functional safety engineers since 2011.

Components that use ML algorithms are specific software modules that interface with other safety-related software components and run on safety-related hardware components, thus they have to be designed, integrated, validated and tested with other components according to the ISO 26262.

Even if components that use ML algorithms are crucial for perception functionality in autonomous driving, this is just one of many vehicle functions.

ISO 26262 does not provide detailed information how to define requirements for concrete functionalities, e.g. it is not specified how to provide a safe persistent storage or safe CPU. Therefore, this is compliant with our approach, as we do not provide a detailed guidance on ML.

A detailed guidance and good practices recommended to ML can be defined in separate documents as ISO 26262 does not provide any methodology how to achieve the required rigour level.

ISO 26262 leaves a lot for interpretation, especially with regard to the used technology or it refers to test methods without defining them in details, e.g. fault injection test.

Working according to V-model defined in ISO 26262 allows to achieve wide-range and bidirectional traceability across item-level hazard analysis to deployment of binary code on hardware platform.

The autonomous driving system in general consists of four modules responsible for: data acquisition (sensors), perception, route planning and steering. ML algorithms are a core of perception module, in which raw information from sensors are transformed into worthwhile information. The remaining modules that are included in an autonomous driving functionality can be development as...
classical components that can be directly designed and developed according to ISO 26262.

The life-cycle of the product development defined in ISO 26262 standard is divided into phases and sub-phases for which the required output work products are defined. In this section, it is pointed out which sub-phases and work products are affected when ML-based components are to be used in a vehicle (using standard designations of ISO 26262 work products) and how a development of a ML-based component for autonomous driving could be incorporated into the V-model proposed in ISO 26262. Additionally, to simplify reading and understanding, we provided a part of exemplary analysis for road line detection and we mentioned the main problems that should be addressed in the respective work products.

In the scope of this work, we focus on supervised learning and deep neural networks as it is generally reported that they achieve superior performance in many computer vision tasks related to autonomous driving [1, 24, 27]. This document does not address problems related to the development of end-to-end learning [2] components, in which DNN are trained on raw sensors data to directly perform steering of the vehicle.

3 - Concept phase

3.5 Item definition

3.5.5.1 Item definition

The objective of this work product is to define properties of the ML-based item equipped at the vehicle and Operational Design Domain (ODD), in which the Autonomous Driving functionality is designed to properly operate (i.e. environmental conditions, background scene, geographic domain, speed limit, etc.). It also needs to address SAE J3016 [20] standard by identifying the automation level.

Example

ITEM: SAE L3 highway pilot.

VEHICLE FUNCTION: vehicle shall keep the lane on a highway without lane changing.

DESCRIPTION OF THE ITEM: Data from radars, lidars and cameras will be used to control engine, brakes and steering.

ODD: driving on a dry highway without rain or snow during daylight with max. speed limit of 120 km/h.

3.6 Hazard analysis and risk assessment

3.6.5.1 Hazard analysis and risk assessment report
The objective of this work product is to define new types of hazards covering functional limitations of ML-based components and related safety goals.

**EXAMPLE**

HAZARD: highway pilot incorrectly localized road lines due to model limitations, e.g. new pattern.

SAFETY GOAL: Prevent crossing the lane.

A good point to start a collection of hazards specific for computer vision domain was introduced in [31].

**3-7 Functional safety concept**

**3-7.5.1 Functional safety concept**

The objective of this work product is to define high-level requirements that are specific for ML components to control of relevant faults in accordance with its safety goals and assign them to the respective elements in the system architectural design.

**EXAMPLE**

FUNCTIONAL SAFETY REQUIREMENTS: The vehicle shall recognize road lines and environment according to the defined evaluation criteria.

ALLOCATION TO ELEMENT: Allocated to sensing subsystem and brain ECU.

**4 - Product development at the system level**

**4-6 Technical safety concept**

**4-6.5.1 Technical safety requirements specification**

The objective of this work product is to define technical safety requirements that specify how to achieve safe execution of ML-based components.

**EXAMPLE**

Camera ECU shall receive raw data from camera sensor, it shall convert, filter, process it and it shall provide it periodically every 10ms over Ethernet to brain ECU that realize the perception functionality.

**4-6.5.2 Technical safety concept**

The objective of this work product is to define technical safety concept that provides an how system architectural design fulfills safety requirements, considering known limitations of ML.

**EXAMPLE**

Road lines shall be localized using two diverse ML algorithms that use a camera. The first method shall work on SoC with GPU, the latter shall work on CPU and voting should be applied between results of both algorithms.

**4-6.5.3 System architectural design specification**

The objective of this work product is to define system architectural design that shall realize the technical safety requirements, considering known limitations of ML algorithms.

**EXAMPLE**

Data provided from lidar sensors shall be in the format of a point cloud.

**4-6.5.7 Safety analyses report**

The objective of this work product is to provide a safety analysis on the system level that verifies each system-level ML component and which demonstrates the independence between independent elements.

**EXAMPLE**

Both ML components use similar DNN and were trained on the same training dataset and their level of diversity may be insufficient, therefore it is recommended to replace one DNN by other type of DNN or classical ML algorithm.

**4-7 System and item integration and testing**

**4-7.5.1 Integration and test strategy**

The objective of this work product is to define a methodology/approach how to test on the target vehicle against technical safety concept considering known limitations of components that used ML algorithms.

**EXAMPLE**

Extensive testing by injecting faults to input for one of SoC to verify if that is detected by the voter defined in TSC.

Extensive testing by providing generated and real images/scenes to the brain ECU.

**4-7.5.2 Integration and test report**

The objective of this work product is to provide test specifications and a report that covers hardware-software integration and testing as well as integration of individual elements of the system on the target vehicle according to the defined specification.

**EXAMPLE**

Extensive tests can find that some objects are not known to Brain ECU.

**4-8 Safety validation**

**4-8.5.1 Safety validation specification including safety validation environment description**

The objective of this work product is to specify ML/AD-related road tests addressing and providing argumentation about their coverage and completeness in the context of known limitations of ML. In the scope of this paper, we do not provide a guidance how this road tests should be defined and how to verify their coverage, but we only argue that this can be done in this work product. Challenges and limitations of road testing are extensively discussed in [15, 16].

**EXAMPLE**

Autonomous vehicle shall be tested on the highway in the defined ODD.

**4-8.5.2 Safety validation report resulting from requirements**

The objective of this work product is to perform the specified road tests, report the results and make decision if the current version of the product ensures expected level of safety.

**EXAMPLE**

Current version of the highway pilot doesn’t recognize bad weather conditions and allows you to enable the autopilot if the weather conditions are not fulfilled.

**6 - Product development at the software level**

**6-5 General topics for the product development at the software level**

**6-5.5.1 Documentation of the software development environment**
The objective of this work product is to select the ML development environment and its configuration.

**EXAMPLE**
Selected DNN for road lines detection shall be implemented in Tensorflow framework. The training of the model shall use Python and the runtime model shall be developed in C++.

### 6-6 Specification of software safety requirements

#### 6-6.5.1 Software safety requirements specification

The objective of this work product is to define general requirements for selected ML algorithm for the dataset. This work product should cover the following aspects:

- specification of the dataset (how much data, what type of data are needed (e.g. object classes, ODD, weather conditions, geographic domain, background scene), how to split data into training, validation and testing);
- specification of the labeling policy (how to annotate the data e.g. how to deal with occluded objects, how many annotators should annotate the same data);
- specification of KPIs for dataset (labeling quality evaluation, dataset coverage, dataset distribution);
- specification for the ML algorithm (what type of algorithm should be used, what exactly the algorithm should do, what performance is required according to the defined KPI, what are the computational complexity is required),
- requirements on KPIs for ML model (metrics to measure single model performance e.g. average precision, measuring robustness of the model against noise, data augmentation, adversarial attacks, reproducibility of the results),
- requirements on runtime monitoring (metrics that will be run on the target vehicle to prevent against potential failures e.g. uncertainty metrics, recognition out-of-distribution data)

The scope of this work product probably will be one of the most challenging due to the fact that our current knowledge about ML and DNN is not yet sufficient. However, specific software safety requirement related to ML algorithms and dataset can be included as a part of this work product. The challenges and limitations related to the dataset design and development were defined in [30] and review of the algorithms related with perception in autonomous driving was presented in [6].

**EXAMPLE**

**DATASET:** The dataset shall contain images gathered for different type of roads at different weather conditions. The road line boundaries shall be marked pixel by pixel. Each image should be annotated by two independent annotators. The amount of 10% of randomly selected data shall be additionally annotated by third annotator. The data acquisition shall take place at the day time.

**MODEL:** Road lines detection should be performed by an algorithm that uses deep neural network and classical computer vision algorithm. Consistency between them should be measured according to the defined KPI.

### 6-6.5.3 Software verification report

The objective of this work product is to verify the requirements defined in the previous work product. This review shall be made jointly by persons responsible for the system, hardware and software development and also ML engineering.

**EXAMPLE**

**DATASET:** During the review, it was determined that requirements:
- specify only one annotator for doing the annotation, which is clearly insufficient.
- do not include acquiring images at night time.
- do not consider data acquisition at different weather conditions.
- do not specify that verification of the dataset completeness shall be performed by a different team than the data acquisition.

**MODEL:** Lack of argumentation for the algorithm selection aspect was identified and this argumentation shall be provided.

### 6-7 Software architectural design

#### 6-7.5.1 Software architectural design specification

The objective of this work product is to design the software architecture for the used ML components including preprocessing of the input data from sensors, selection of the ML algorithms and data flow between them, and postprocessing of the outputs of ML algorithms. It shall also define architectural design of the ML model training, its integration with KPI to evaluate the performance and robustness of the model during training. Additionally, the architecture shall also address the problem of integration of the ML model with runtime monitoring metrics, which allows to supervise the ML models behaviour (e.g. analysis of the neuron activation patterns [5]) or correctness of the input data (e.g. using Monte Carlo epistemic uncertainty to measure how the analyzed sample is statistically similar to these samples that were used in the training dataset [13]) at inference time.

This part should also cover architectural design of the dataset development process and specify how the data should be recorded, collected and annotated (format of the data, compression, labels specification). The problem of architectural design definition for data gathering and labeling process is typically neglected, but in our opinion these steps shall be considered as a part of the product development. One of the possible approaches for handling these aspects could be using active learning [3, 18] in the development life cycle to label only these data samples for which it is expected that they improve quality of the model or to tune the model to deal with edge cases that were not recognized during the tests.

**EXAMPLE**

**DATASET:** Road line boundaries annotations shall be stored as xml files separate for each input image. The bounding box specification shall define coordinates as $[y_{min}, x_{min}, y_{max}, x_{max}]$ format, where $y$ and $x$ are corresponding to the bounding box height and width measured in pixels.

**MODEL:** Road lines shall be localized using LaneNet network [28] and Canny detector [29] and consistency between them should be measured according to the defined KPI to verify if road lines were annotated correctly.
are fulfilled and the all safety-related parts of the software were identified. In scope of this work product safety measures shall specified to mitigate potential failures related to ML. Additionally, it should deliver the definition of verification criteria to confirm effectiveness of designed safety measures.

**EXAMPLE**

**DATASET:** Incorrect road line annotation could lead to decrease of the final model performance.
**MODEL:** A corrupted input image provided by a software communication layer to the DNN could cause similar effect to the adversarial attack on the model.

**6-7.5.3 Dependent failures analysis report**
The objective of this work product is to analyze if the implementation of software safety requirements defined for ML components ensures independence or freedom from interference and how possible dependent failures can be mitigated.

**EXAMPLE**

**DATASET:** It was identified that the second set of annotation was prepared by translating the first set of annotations by a constant offset, so that it is doubtful that both annotations set were prepared independently.
**MODEL:** Two models use the same input camera sensor and therefore corruption of that sensor might cause a failure of both DNNs. Both algorithms are developed using the same ML framework and thus the achieved level of diversity is not sufficient (e.g. due to a potential systematic fault in the implementation of a certain DNN layer that is commonly used).

**6-7.5.4 Software verification report**
The objective of this work product is to verify the correctness of the developed software architectural design and provide the evidence that it fulfills software safety requirements considering known limitations of ML.

**EXAMPLE**

**DATASET:** The performed inspection of the architectural design confirmed that it fulfills all software safety requirements.
**MODEL:** The performed inspection revealed that the connection between epistemic uncertainty estimation and the runtime model is not yet defined.

**6-8 Software unit design and implementation**

**6-8.5.1 Software unit design specification**
The objective of this work product is to specify the functional behaviour and the detailed information that are necessary for implementation of ML algorithms and KPIs and runtime monitoring. It comprises unit design of the host and target frameworks (that are typically preexisting). It should also include all internal models’ parameters and hyperparameters and their optimization during the training and validation steps. This part should be continued until the models achieve the expected effectiveness. Finally, the output of this work product should be frozen models (e.g. a deep neural networks with the weights), which will be not changed or modified anymore. This part also shall cover the data acquisition and annotation process. As was discussed for software architectural design, in our opinion it is crucial to include dataset development to the V-model defined in ISO 26262 to define and implement how the process of data collection is integrated with the training of the model. The output of this work product shall be a dataset with annotations that are divided into training, validation and testing subsets.

**EXAMPLE**

**DATASET:** 60% of the gathered data shall used as the training, 20% as validation and 20% as testing.
**MODEL:** The host model shall be implemented using TensorFlow framework. The target model shall be generated from the host model and executed via dedicated execution environment. The optimal parameters and hyperparameters defined for the LaneNet model shall be established in training and validation process to fulfill defined values of KPI considering robustness of the model against data augmentation, impact of different noise models and adversarial attacks.

**6-8.5.2 Software unit implementation**
The objective of this work product is to implement the required ML algorithms (both host and target), perform training and validation of the selected models, optimize their internal parameters and evaluate of KPI and runtime monitoring metrics. In this part the data shall be gathered and annotated. The results of this step shall be frozen ML models that are not to be changed in software unit testing. Obviously, until a subset of the annotated data is not provided, it is not possible to start the training of the models, but it could be realized as iterative and parallel process while more and more data samples are acquired and labeled.

**EXAMPLE**

**DATASET:** Gathering and annotation process of the images with road lines.
**MODEL:** Implementation, training and validation of the LaneNet network to correctly detect the road line according to the defined KPI. Generation of the target model from the host model.

**6-9 Software unit verification**

**6-9.5.1 Software verification specification**
The objective of this work product is to specify how to test the final model on the test dataset and according to which metrics to achieve the expected performance. This work product shall also specify how the quality of the labeling process should be verified.

**EXAMPLE**

**DATASET:** 20% of the data shall be send to other supplier to check if all road lines were annotated correctly.
**MODEL:** To evaluate the reproducibility of the model, the 10 independent LaneNet models shall be tested.

**6-9.5.2 Software verification report**
The objective of this work product is to test frozen models (obtained during the training phase) on the test dataset according to the defined KPI, measuring the performance of implemented runtime monitors and perform standard verification testing recommended by ISO 26262. In the scope of this paper, we do not discuss which testing techniques that are applicable to ML algorithm, but we would like to address that this verification should be applied in this work product on the model that were obtained after the training process was finished. Additional important part of this work product is also back-to-back testing between training
model and runtime model. This work product shall also cover verification of the quality of the labelling process. Finally, this work product shall also cover whether the quality of the dataset labelling meets the defined requirements.

**EXAMPLE**

DATASET: Verification if the road line boundaries were annotated pixel-by-pixel.
MODEL: Verification if defined runtime monitors are able to identify so-called edge cases that could cause incorrect output from the network (e.g., highway pilot is not designed to work during the rain, but designed runtime monitor incorrectly recognizes ODD).

**6-10 Software integration and verification**

**6-10.5.1 Software verification specification (refined)**
The objective of this work product is to specify how to test the wider scope of the system with both ML networks integrated according to selected KPI and runtime monitoring metrics. This work product shall also specify how to verify the consistency between two subsets of annotations provided by independent annotators or suppliers.

**EXAMPLE**

DATASET: Consistency between two independent road line annotations obtained from different suppliers shall be evaluated by calculating inter-annotator variance.
MODEL: Performance of the two integrated road lines detectors: the LaneNet network and Canny edge detector shall be evaluated according to defined KPI and runtime monitors.

**6-10.5.2 Embedded software**
The objective of this work product is to define the steps that are required to integrate ML models implemented at software unit design level into bigger ones. Considering the dataset, in this work package, it should be defined how to integrate the annotations that were generated from many annotators or suppliers and what kind of voting model is to be utilized.

**EXAMPLE**

DATASET: The coordinates of the manually labelled road lines generated by two annotators shall be averaged.
MODEL: A voter should check whether a designated road lane was based on road lines detected on at least one model.

**6-10.5.3 Software verification report (refined)**
The objective of this work product is to test the performance of the integrated ML models according to the defined KPI and runtime monitors. Additionally, if the dataset annotations were prepared by independent annotators or suppliers then this work product shall also provide a consistency verification report. Finally, this work product should give the answer if the consistency between dataset labelling annotations and integrated ML models is sufficient or the software architectural design should be updated.

**EXAMPLE**

DATASET: A report from evaluation of the consistency of two subsets of annotations provided by two independent suppliers.
MODEL: A report that summarizes performance the integrated road line detection models: LaneNet and Canny edge detector.

**6-11 Testing of the embedded software**

**6-11.5.1 Software verification specification (refined)**
The objective of this work product is to specify the testing of entire software including runtime monitoring on the target vehicle. This part shall also define criteria to test whether developed ML component fulfils the software safety requirements on the target vehicle. Additionally, this part shall define how to test if all software safety requirements defined for the dataset were fulfilled. Finally, this work product shall specify how to test the integration of the whole process of data collection, labeling and training of a new model in case runtime monitors detect edge cases that were incorrectly recognized by currently developed ML component.

**EXAMPLE**

DATASET: Specification how to test if dataset consists of the images gathered in winter conditions.
MODEL: Specification of E2E testing of the road line detection component.

**6-11.5.2 Software verification report (refined)**
The objective of this work product is to perform the final tests of ML based component and annotated dataset according to the specification defined in the software verification specification to finally confirm that they fulfill all allocated software safety requirements.

**EXAMPLE**

DATASET: The road lines detection dataset is not complete, because it does not contain example of highway roads recorded during winter conditions and it shall be updated.
MODEL: The used runtime monitor designed for road line detection component is not able to identify winter condition and allows to turn on highway autopilot in winter conditions.

**7 - Operation, Service and Decommissioning**

**7-7 Operation, service and decommissioning**

**7-7.5.1 Field observation instructions**
The objective of this work product is to perform a continuous monitoring of released vehicles after the start of production. A field monitoring process shall be available that allows for a possibility to detect outdated ML-models or anomalies (e.g., out-of-distribution detection) and the reaction on defined trigger events for (planned) evolutionary updates over the whole safety lifecycle. The field monitoring process shall especially be planned to determine what, when, by whom and how often data shall be processed and updated. Field monitoring and change management shall be closely interconnected.

**EXAMPLE**

The out-of-distribution detection running in a vehicle detects anomalous events that require a degradation of the ML-based function.

**8 - Supporting processes**

**8-8 Change Management**

**8-8.5.1 Change management plan**
The objective of this work product is to specify a change management plan that applies for a project that is under continuous
delivery (rather than when there is just one delivery - start of production). ML-based systems will very likely have to be updated continuously over the product lifetime. The change management planning therefore shall consider all foreseeable trigger events that possibly imply a change, such as explicitly planned continuous changes, changes necessary due to detected anomalies or due to the aging of demands. This analysis and planning activity shall be performed at an early stage of the development. Having a mature change management planning available will be essential to automate activities by incorporating the backend tool chain.

EXAMPLE

Aging of demands: Due to a change in legislation, new test cases are mandatory which leads to an update (re-training) of the ML-based system.

8-11 Confidence in the use of software tools

8-11.5.1 Software tool criteria evaluation report

The objective of this work product is to evaluate the impact of the ML tools, used during the development process, on the released product. In a ML-based project, the ML-tools are more complex than in C development. Secondly, the errors that they introduce (e.g. generators, training) or errors that they detect (ML simulation or testing tools) are hard to be detected by other means, so potentially there is a risk of having several complex tools that have a Tool Confidence Level of 2 or 3.

EXAMPLE

Because of usage of two diverse development frameworks (PyTorch and TensorFlow) and due to testing of the learned networks, the frameworks (running at host only) are classified as TCI1.

8-11.5.2 Software tool qualification report

The objective of this work product is to perform a safety-qualification of ML tools. Such tools are complex, they are not developed according to safety standards.

EXAMPLE

A selected TCI2 test framework has a good script-based development process, the software is deployed in thousands of hosts in the evaluated major version, there are extensive unit tests for the tool as well as an independently developed validation test framework covering all safety-related features of the tool, which all allows to achieve TCI2 for this tool.

Defining the whole process according to the workflow proposed by ISO 26262 makes it possible to easily trace potential failures and limitations in the product development. For instance, in the considered example of a road lane detection, we can discover that the currently used algorithm is not able to correctly detect a road line if it is not marked by white lines, because the acquired dataset does not contain such type of data in the training part.

4 CONCLUSIONS

In this paper, we propose how the ML based components can be currently developed according to the ISO 26262 standard. Our work is mainly motivated by the fact that still there is a lack of dedicated standard that covers issues related to the design and development of ML based components in road vehicles.

By this work we also try to prove that design and development after some adaptations can be started based on existing version of the standard. Defining the whole process according to the workflow proposed by ISO 26262 makes it possible to easily trace potential failures and limitations in the design and development process of autonomous driving and can be applied immediately before a dedicated standard is be developed. We believe that our work will be a contribution to further progress in application of ML in automotive industry.

REFERENCES

[1] Claudeine Badue, Rànik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius Brito Cardoso, Avelino Forechi, Luan Ferreira Reis Jesus, Rodrigo Ferreira Berriel, Thiago Meireles Paziax, Filipe Mutz, Thiago Oliveira-Santos, and Alberto Ferreira De Souza. 2019. Self-Driving Cars: A Survey. arXiv:1901.04407

[2] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jaksa Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. 2016. End to End Learning for Deep Object Detection. arXiv:1609.09875

[4] Chih-Hong Cheng, Chang-Hao Huang, and Georg Nürehnberg. 2018. nn-dependability-kit: Engineering Neural Networks for Safety-Critical Systems. CoRR abs/1811.06746 (2018). http://arxiv.org/abs/1811.06746

[5] C. Cheng, G. Nürehnberg, and H. Yanoka. 2019. Runtime Monitoring Neural Activation Patterns. In 2019 Design, Automation Test in Europe Conference Exhibition (DATE) 300–303.

[6] Di Deng, Christian Haase-Schnett, Lars Rosenbaum, Heinz Hertlein, Claudius Gläster, Fabian Tunn, Werner Wiesbeck, and Klaus Dietmayer. 2019. Deep Multimodal Object Detection and Semantic Segmentation for Autonomous Driving. Datasets, Methods, and Challenges. arXiv:1902.07830

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Conference on Computer Vision and Pattern Recognition (CVPR).

[8] J. Henriksson, M. Borg, and C. Englund. 2018. Automotive Safety and Machine Learning: Initial Results from a Study on How to Adapt the ISO 26262 Safety Standard. In 2018 IEEE/ACM 1st International Workshop on Software Engineering for AI in Autonomous Systems (SEFAS). 47–49.

[9] Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xiping Yi. 2018. A Survey of Safety and Trustworthiness of Deep Neural Networks. arXiv:1812.08342

[10] IEC. 2010. IEC 61508 Part 1-7 Second Edition.

[11] ISO. 2018. ISO 26262 (Part 1-12) – Road vehicles – Functional Safety, Second edition. http://www.iso.org

[12] ISO. 2019. ISO/PAS 21448 – Road vehicles - Safety of the intended functionality, First edition. http://www.iso.org

[13] Alex Kendall and Yarin Gal. 2017. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS ’17). 5580–5589.

[14] Philip Koopman and Beth Osyk. 2019. A Safety Standard Approach for Fully Autonomous Vehicles. In Computer Safety, Reliability, and Security, Alexander Romanovsky, Elena Troubitsyna, Ilir Gashi, Erwin Schotsch, and Friedemann Bötsch (Eds.). 326–332.

[15] Philip Koopman and Beth Osyk. 2019. Safety Argument Considerations for Public Road Testing of Autonomous Vehicles. SAE Int. J. Adv. & Curr. Prac. in Mobility 1 (04 2019), 512–523.

[16] Philip Koopman and Michael Wagner. 2016. Challenges in Autonomous Vehicle Testing and Validation. SAE Int. J. Trans. Safety 4 (04 2016), 15–24.

[17] C. B. S. T. Molina, J. R. d. Almeida, L. F. Vismari, R. I. R. González, J. K. Naufal, and A. M. B. M. Ferreira De Souza. 2019. Self-Driving Cars: A Survey. arXiv:cs.CV/1604.07316

[18] [19] F. Rudzicz, PA. Paprica, and M. Janczarski. 2019. Towards international standards for evaluating machine learning. In Safelab at AAAI19.

[20] [21] Rick Salay and Krzysztof Czarnecki. 2018. Using Machine Learning Safely in Automotive Software: An Assessment and Adoption of Software Process Requirements in ISO 26262. abs/1808.01614 (2018). http://arxiv.org/abs/1808.01614
[22] Rick Salay and Krzysztof Czarnecki. 2019. Improving ML Safety with Partial Specifications. In Computer Safety, Reliability, and Security, Alexander Romanovsky, Elena Toubiotsyna, Ilo Cushi, Erwin Schoitsch, and Friedemann Bitsch (Eds.). Springer International Publishing, Cham, 288–300.

[23] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. 2018. An Analysis of ISO 26262: Machine Learning and Safety in Automotive Software. In SAE Technical Paper. SAE International.

[24] Saptarshi Sengupta, Sanchita Basak, Pallabi Saikia, Sayak Paul, Vasilios Tsalavoutis, Frederick Atiah, Vadlamani Ravi, and Alan Peters. 2019. A Review of Deep Learning with Special Emphasis on Architectures, Applications and Recent Trends. arXiv:cs.LG/1905.13294

[25] B. Spanfelner, D. Richter, S. Ebel, U. Wilhelm, W. Branz, and C. Patz. 2012. Challenges in applying the ISO 26262 for driver assistance systems. Tagung Fahrerassistenz (2012).

[26] Burkhard Stiller, Thomas Bocek, Fabio Hecht, Guilherme Machado, Peter Racz, and Martin Waldburger. 2019. Safety First for Automated Driving. Technical Report.

[27] V. Sze, Y. Chen, T. Yang, and J. S. Emer. 2017. Efficient Processing of Deep Neural Networks: A Tutorial and Survey. Proc. IEEE 105, 12 (2017), 2295–2329.

[28] Ze Wang, Weiqiang Ren, and Qiang Qiu. 2018. LaneNet: Real-Time Lane Detection Networks for Autonomous Driving. arXiv:1807.01726

[29] Pei-Chen Wu, Chin-Yu Chang, and Chang Hong Lin. 2014. Lane-mark extraction for automobiles under complex conditions. Pattern Recognition 47, 8 (2014), 2756–2767.

[30] O. Zendel, K. Honauer, M. Murschitz, M. Humerberger, and G. F. Domínguez. 2017. Analyzing Computer Vision Data — The Good, the Bad and the Ugly. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 6670–6680.

[31] O. Zendel, M. Murschitz, M. Humerberger, and W. Herzner. 2015. CV-HAZOP: Introducing Test Data Validation for Computer Vision. In 2015 IEEE International Conference on Computer Vision (ICCV). 2066–2074.