DL-SPICE: GUIDELINES FOR AI/ML COMPONENT SPECIFICATION

VER 3.0

MARCH 2023

USCAR
UNITED STATES COUNCIL FOR AUTOMOTIVE RESEARCH

AI/ML V&V Workgroup

Workgroup Members[1]

Dalong Li, Formerly Stellantis Inc.
Ramesh S, General Motors
Devesh Upadhyay, Ford

[1] The authors' names appear in alphabetical order of their last name.

Executive Summary

Next generation vehicles are primed for increased use of features involving Artificial Intelligence (AI) /Machine Learning (ML) functions for many Advanced Driver Assist Systems (ADAS) and Autonomous Vehicles (AV). These features, often involving perception, planning or vehicle behavior functions, rely on very high dimensional inputs in the form of unstructured data objects produced by sensors like camera, LiDAR and Radar to deliver a variety of functions like lane marking identification, traffic sign and signal classification, vulnerable road user detection, etc. The design, development and validation of these functions is fundamentally different from traditional software developed using well defined requirements and algorithmic procedures. While there are well established and agreed upon standards and guidelines for providers of traditional software (SW) components, no common guidelines are being followed by the suppliers of AI/ML based components and it is becoming difficult for the OEMs to validate and accept the supplier provided components involving AI/ML functions. Further, given the fundamental differences of AI/ML components from traditional SW, there is a need for rethinking robustness, testing and Verification and Validation (V&V) of these components. Additionally, given the growing numbers of providers in this new space there is an urgent need for the automotive industry to develop a consolidated set of guidelines related to the development, performance verification and life-cycle management of AI/ML related components.

It is in this context that this workgroup has been working on drafting a set of guidelines for assisting the development and V&V of these AI/ML components that are typically used in the context of perception and planning subsystems in next generation ADAS/AV features. For this work, the team drew inspiration from Automotive Software Process Improvement and Capability dEtermination (ASPICE). ASPICE defines the best practices for the development of Automotive grade embedded SW. For the purposes of providing similar guidelines for the development of AI/ML software the term DL-SPICE was adopted where DL refers to “Deep Learning”. DL-SPICE, similar to ASPICE, is intended to grow into an automotive standard for V&V of AI/ML components. DL-SPICE primarily identifies all assets that are to be developed as part of the functional AI/ML component and defines detailed development guidelines, methods and processes for V&V and life-cycle management of the AI/ML component.

Table of Contents

Introduction

Recent advances in Artificial Intelligence (AI) and Machine Learning (ML) such as Deep Neural Networks (DNN) and Reinforcement Learning (RL) are fundamentally transforming industries and businesses. The Automotive industry is no exception. Advanced driver-assist systems (ADAS), and general autonomy features already employ advanced perception and decision-making techniques that depend upon AI/ML components. Even though the Validation and Verification (V&V) of traditional automotive systems is very well understood it still consumes a significant amount of development time and effort. AI/ML components are fundamentally stochastic in nature and more complex relative to traditional SW. This, combined with the anticipated increase in the adoption of AI/ML components, poses new challenges for the automotive industry. The V&V of these systems (especially safety critical functionality of these systems) is expected to become more complex and more resource intensive hence a systematic approach for robustness verification of these components is essential. For the purposes of this document, we refer to traditional embedded software as SW1.0 and all embedded AI/ML software as SW2.0, a term first used by A Karpathy in [1]

There are several challenges for implementing verification and validation of AI/ML components. Some of these are listed below:

- AI/ML components often lack precise requirements or specifications mapped to the intended operating domain.
- AI/ML components are inherently stochastic and, on many instances, their responses/predictions may not be well understood, this is also very often referred to as the lack of interpretability of AI/ML components.
- Like traditional software, V&V of AI/ML components requires knowledge (design and implementation details) of the SW. But in addition, unlike tradition SW, the V&V of AI/ML components also requires details on data, architecture and methods used during training and testing in the development phase.
- Since ML models are data driven, any associated drifts in the model and/or training data post deployment make a one-time V&V of an AI/ML component inadequate. ML models therefore require continuous monitoring (CM) post deployment. This extends the CI/CD process used in traditional SW to CI/CD/CM and such a process must be defined for all deployed AI/ML components.

- AI/ML model developers typically use open-source libraries and/or development platforms, hence a standardization of libraries and platforms proven in practice, is required for deployment.
- The use of open-source data and algorithms may very often require commercial licensing and usage compliance hence transparency in the use of such resources is essential.

Given these challenges, it is unclear as to how the existing processes methods for SW1.0 may be extended to AI/ML based systems and SW2.0. Therefore, development and V&V guidelines, that allow a systematic demonstration of robustness of the AI/ML components are necessary. This is a major focus of an initiative kicked off through the United States Council for Automotive Research LLC (USCAR). The objective of this workstream is to ideate around guidelines, processes, and requirements for the development, testing and V&V of AI/ML based systems.

USCAR is the collaborative automotive technology company for Ford Motor Company, General Motors, and Stellantis N.V (Chrysler). The goal of USCAR is to further strengthen the technology base of the domestic auto industry through cooperative research and development.

Through USCAR, members can collectively tackle problems of common interest while achieving significant efficiencies. These research tasks would be far more difficult, and in many instances, impossible to achieve as quickly by individual companies alone.

USCAR collaborations often expand to include external research partners and associates. These can include R&D with emerging industries, codes and standards organizations, automotive suppliers, universities, energy companies and utilities, government agencies and national laboratories.

Motivation & Overview

The focus of this document is to define a set of guidelines and principles for automotive grade AI/ML components that are being deployed in the perception and planning subsystems of features of Level 2 or above. These guidelines are expected to assist all providers of AI/ML components, whether internal to an OEM or external resources such as third-party vendors. This is especially critical for creating a level of consistency between external suppliers of such components. These guidelines aim to provide sufficient information for exercising and assessing the verification and validation and robustness quantification of AI/ML components. One main purpose of these guidelines is to identify additional artefacts, e.g., V&V results, data set details, architecture details, etc. to be made available by the suppliers along with the AI/ML components. These additional artefacts should provide reasonable evidence and enable OEMs to make the necessary assessment on the performance quality of the component. These guidelines also indicate the necessary artefacts, e.g., requirements, feature context, etc., to be provided by the OEMs to the suppliers developing the component.

In developing these guidelines, the team was largely inspired by Automotive SPICE standard (ASPICE) that defines the best practices for the development of Automotive grade embedded software [2]. ASPICE is built on top of Software Process Improvement and Capability dEtermination (SPICE) which is a framework for general software process assessment developed by the ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission). Its purpose is to evaluate development factors that allow assessors to determine an organization's capacity for effectively and reliably delivering software products.

ASPICE guidelines were found to be insufficient for AI/ML components due to the nature of these components. Unlike traditional software, the development of AI/ML components is driven in large part by training data sets and less so on explicit verifiable rules; further they involve stochastic optimization algorithms that employ complex heuristics. Further the inputs to an AI/ML model are often high dimensional, may exhibit high variability and inherent invariances that very often compromise the robustness of an AI/ML component.

Inspired by ASPICE, the proposed guidelines are referred to as DL-SPICE where DL refers to “Deep Learning”. DL-SPICE, similar to ASPICE is intended to grow into an automotive standard for software development and V&V for AI/ML components. Like ASPICE, OEMs can use the proposed DL-SPICE framework to assess the capabilities of any process used (by providers) in the development of AI/ML components. OEMs may also define their internal development process to be DL-SPICE compliant. Figure 1 shows an adaptation of the ASPICE Design-V architecture for developing and deploying AI/ML based components. A major departure from the ASPICE framework is reflected in the multiple iterative loops that may become necessary given the nature of AI/ML models requiring continuous monitoring and performance adjustment including re-training. Another significant difference is the addition of data management phase which is unique to AI/ML component. The development of the dataset used in the training and validation of the AI component assumes greater importance and requires careful consideration. Figures 2 and 3 further define the various terms used in the DL-SPICE framework.

Many modern driver-assist features, e.g., Automatic Lane Centering, Automatic Emergency Braking Systems (AEB), can benefit from using AI/ML components in their perception tasks. Such a perception task takes vision data from camera, Lidar, Radar and other vision-based sensors and process them using modern machine learning algorithms to produce usable information such as classification, object detection, tracking, etc. that are then used for some smart actions such as controlling the vehicle motion. Figure 2 provides a very simplified view of AEB feature that uses a perception-based AI/ML component. The AI/ML component detects potential objects on the road from the various sensors and provides inputs to the vehicle motion control subsystem.

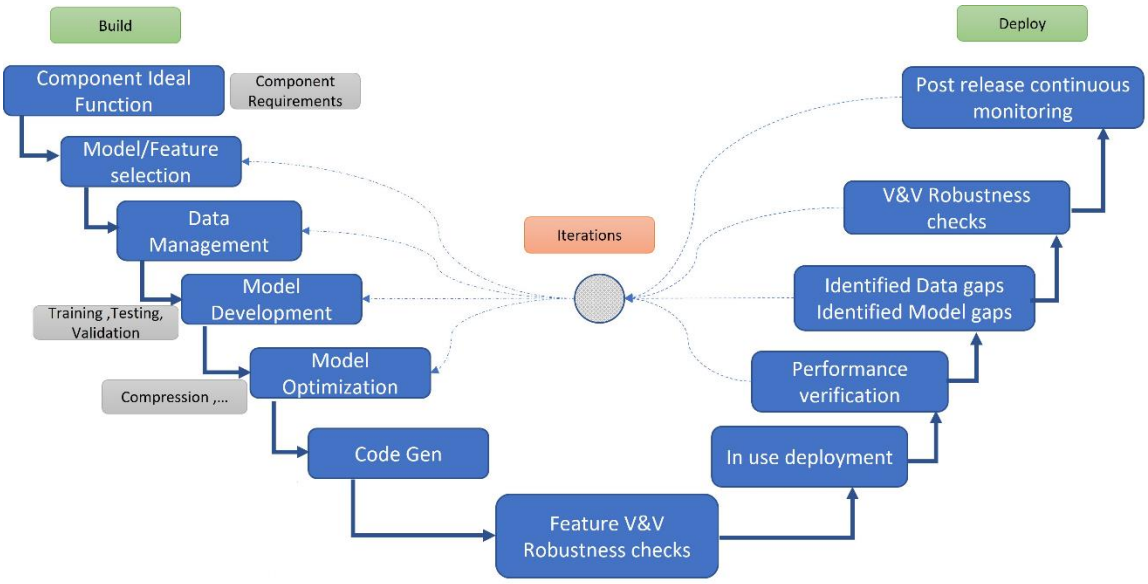


Figure 1: A proposed DL-SPICE workflow

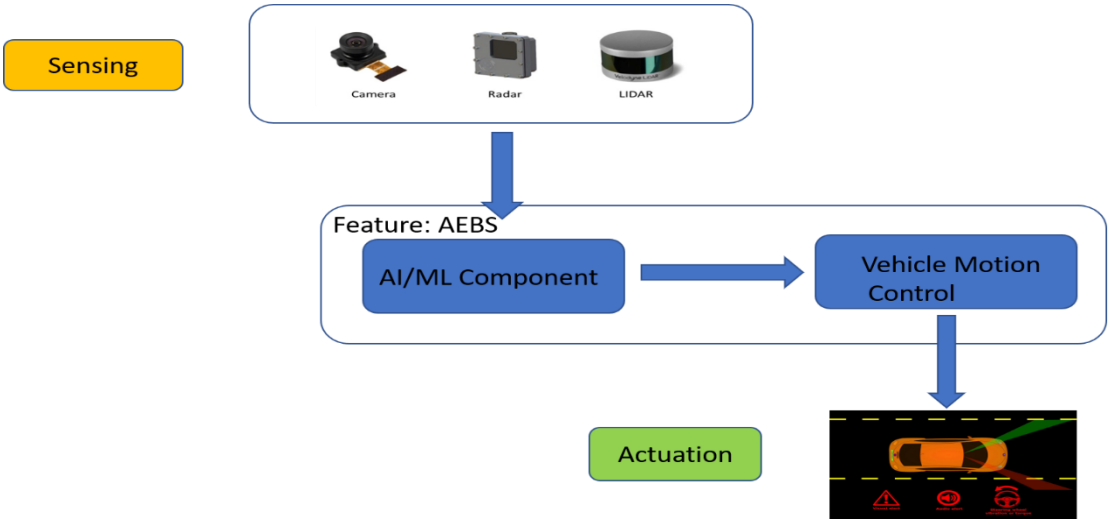


Figure 2: AI/ML based application

An AI/ML component typically employs a set of parameters whose values determine its behavior. The values for these parameters are `learnt' (determined) as part of the development of the component through a series of iterative training steps using a large number

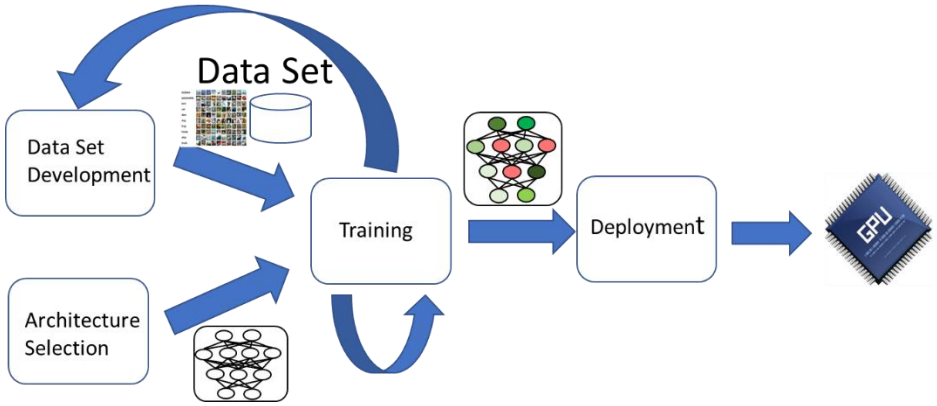


Figure 3: AI/ML Component Development Steps

of example inputs and the expected outputs (also known as ground truth). Unfortunately, there are several factors that contribute to the uncertain or non-robust and often unintended behavior of these components requiring additional V&V safeguards. These include the complexity of the inputs and the wide variability (e.g., different lighting conditions, noises and distortions, etc.), the human based semantic nature of the functionality, e.g., traffic signs, pedestrians, etc., and the iterative learning/relearning steps. It is assumed that there is no direct online learning capability of these components. Figure 3 highlights the typical steps involved in the single iteration of the development of an AI/ML component. These steps are repeated during the development to meet the appropriate requirements as well as post deployment when new performance insufficiencies are discovered. This may require development of continuous monitoring components and including them as part of the AI/ML component in the system.

Guidelines Scope

In this document we focus mainly on AI/ML components using Deep Neural Networks (DNNs). The different steps in the development life cycle of AI/ML components are specific to DNN development. We believe that the guidelines would be helpful and could be extended to other types of AI/ML components.

Another focus of these guidelines is functionality and performance of AI/ML components that have a bearing on the quality and robustness of the component. We are aware of several ongoing national and international efforts (SAE and ISO [3]) on developing guidelines and standardization of AI/ML components that focus on safety and security; some of us are indeed

members of these committees and plan to contribute to those efforts as well. The plan of this proposal is to complement other parallel efforts and to be consistent and synergistic to the guidelines as and when they are released.

Guidelines Specification

The proposed guidelines recommend the use of a specification template to be used by the component developer for providing information regarding the AI/ML component. The specification template consists of multiple sections where each section corresponds to a distinct development or V&V activity and suggests potential development details (artefacts) to be provided.

1. Name and Functionality of the Component:

A descriptive name and high-level description of the ideal function/s of the component being developed are to be provided as outlined in the recommendations below: A descriptive name of the component indicative of the intended functionality (e.g. road-sign detection, pedestrian detection) including any broader feature context.

- The type of prediction (e.g. classification, detection, regression, tracking, decision, etc.,)
- A high-level description of all the inputs. The input definition contains information about the type of data/information consumed by the component, such as:
    - Image type: RGB, RGB-D, Point-cloud
    - Image size in pixel grid
    - Channels: RGB
    - Frames per Second
    - Mono/Stereo
    - Field of View
    - Lens type
    - Sensor mounting
- A high-level description of all outputs and any confidence scores if applicable
    - Ex. Steering angle at confidence score of -0.90
- Sensor type and sensor details including any sensor calibration parameters and calibration methods. This information can be used within internal simulation platforms for simulation based V&V.
- An example camera specification could read like this:
  Image sensor: 1/4"" color CMOS, Lens: 3.6MM
  Infrared LEDS : 10PCS, Pixel :300000
  Night vision : 10M, Mini. Illumination : 0.1Lux
  Image Compression: MJPEG, Image Frame Rate: 30fps, Resolution: 720p
  Network: by cable or Wi-Fi
  Supported Protocol: TCP/UDP/IP/ICMP/DHCP/DNS/HTTP/FTP/SMTP/NTP/UPNP/DDNS/RTSP
  Power: DC 5V 2.0A, Power consumption: 6W/4W (Infrared on/off)
  Dimensions : 200mm x 110mm x 160mm (L x W x H)

2. Component Requirements (OEM + Supplier):

A detailed description of the functional and performance specification of the AI/ML component must be provided in this section. This would be based upon the intended functionality or requirements provided by the OEM. The specification may also include any safety ratings as well as any associated regulatory and compliance mandates. The details may include additional details of any V&V methods outlined by the OEM and the supplier's ability to perform these checks as defined including any deviations.

The following subsections highlight the details to be included in this section.

2.1 Functional Specifications:

2.1.1 Sensor Specification: standardized specification of cameras and other sensors used by the AI/ML component. Sensor specifications includes range, noise factors, illumination, reflectivity constraints, contrast, ideal weather conditions where the sensors are expected to perform. An example specification of the sensor would be:

> *The camera shall be installed on the wind shield, facing forward. It will detect and classify both stationary and moving vehicles (car, truck) on highways and local roads under all weather conditions up to 100 meters. The minimal illumination is 0.1 Lux. In terms of image quality/degradation, Peak signal-to-noise ratio (PSNR) of the image shall be 30dB or higher. Motion blur shall be less than 20 pixels and out of focus blur shall be less than 10 pixels. The detector runs at 10 fps.*

2.1.2 Sensor Location details: The mounting locations of the sensors and the fields of view (FOV) and range for the given locations. Any assessments on the impacts of mounting location on range and FoV must be provided.

2.1.3 Sensor pre-processing: Details of image processing operations performed on the raw images before they are fed to the machine learning component.

2.1.4 Detailed Input-Output Specification:

- description of data after pre-processing, that are input to the AI/ML component; this includes type, dimensionality and range information.
- the expected outputs of the AI/ML component.
- description of the tasks performed by the component, e.g., classification, regression, etc.

2.1.5 Assumptions on the Input Space: The AI/ML component makes assumptions on the syntactic and semantic features of the input objects, e.g., size, color, type,

and shape of the objects in the input fed to the components. These assumptions need to be included as part of the specification.

2.1.6 Functional Noise Factors : where possible, identify any known/expected noise factors associated with the inputs to the sensors, such as:

- lighting: sun flares, darkness, shadows
- environmental factors: foliage, occlusion,
- weather conditions: snow, rain etc.

The impact of these noise factors on the outputs of the AI component must be quantified under a well-defined range of noise uncertainties. While it is also desirable to test against edge cases, it is recognized that generating edge case examples is not always easy for ADAS systems. It is therefore recommended that any known edge/corner cases in the field be considered for V&V if possible.

2.1.7 Functional Adversarial or negative examples: Besides positive examples, it is desirable to include negative examples, especially for classification functions, e.g., cases in which there is likely to be misclassifications. The examples include, plastic bags filled with air or water, fake speed bumps/potholes, banners with human pictures and traffic signals.

2.1.8 Performance Requirements: The compute performance of the component depends upon the inference engine implemented in software and the hardware platform on which it runs. The performance will be gauged by metrics such as accuracy, Area under the curve (AUC), latency, throughput, memory and energy usage as well as any others specified by the OEM. Any suitable metrics developed by the vendor for the specific application may also be included. The use of standard benchmarking platforms such as MLPerf is also encouraged. Further, any expected level of performance degradation on all metrics are to be included, e.g., maximum-delay, minimum throughput, etc.

2.1.9 Non-Functional Requirements: Fault-tolerance capabilities of the components including failure mode detection and avoidance, are to be given. For instance, if there is dust on the camera lens, the camera is blocked, or the camera is impacted by glare then the expected behavior of the component under these noise factors, along with any fault mitigation capabilities are to be included. Mitigation may also include alerting and warning mechanisms. If no mitigating actions are provided, this must be mentioned.

2.1.10 V&V Requirements: This may include the following:

- any feature performance metrics used along with their expected target values, some examples are: confusion matrix, accuracy, recall, IOU percentage, etc.
- any feature robustness metrics and methods used to demonstrate robustness. Methods may include some combination of realistic simulation and/or field investigations. Robustness for both functional and non-function requirements must be demonstrated.

- any training, validation and test coverage requirements agreed upon with the OEM, over any other standard/ mandatory regulatory requirements

The OEM may provide a sample test data set comprising of inputs and corresponding expected outputs of the component against which the V&V may be performed. The vendor is also expected to share any additional data used in performing V&V in support of the results provided to the OEM.

3. Feature Context:

The AI/ML component may provide inputs to a vehicle feature that controls the vehicle motion, hence a feature context diagram along the lines of Figure 2, must be provided. The feature context needs to contain details of the sensor/s that provide input data to the AI/ML component and the output/s of the component that may influence various aspects of the feature.

3.1 Feature Operation Design Domain (ODD): The ODD, according to SAE J3016, defines the operating conditions under which a feature is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic conditions. The ODD includes information related to roadways, zones (school, hospitals), road types, weather, lighting, traffic, time of day, etc. The ODD directly influences both the sensing requirements as well as the performance metrics of the ML component and must be considered when cascading requirements and outlined in this section.

3.2 Feature Object Event Detection & Response (OEDR): A description of the objects and events expected to be detected by the feature and the details of the expected response. Examples include stopping, turning, exits, lane changes etc. OEDR is influenced by traffic dynamics and must therefore also include the response to an event. This places an end to end (sensing to actuation) response time requirement on the feature. Some extensions to the concept of OEDR are found in discussions around behavioral competencies. For instance, Wayve.AI [5], considers three categories of BCs:

- **3.2.1** tactical non-OEDR, e.g. lane keeping
- **3.2.2** tactical OEDR, e.g. detect and respond to static obstacles in the vehicle's path
- **3.2.3** failure mode behavior, e.g. Minimal Risk Manuever MRM

3.3 Robustness Requirements of the Feature: The speed of the vehicle under which the feature is expected to operate may impose some robustness requirements on the component, like motion blur in perception-based systems. The ODD of the feature may decide the types of noises impacting the inputs to the component. Adversarial

attacks on the inputs are a possibility and their impacts on the components and the features must be considered.

3.4 Safety criticality of the feature: The feature may have some safety critical requirements that might impose stronger requirements on the component. These might include safety requirements to address hardware and software failures as well as functional insufficiencies. Recent versions of ISO 26262 and ISO 21448 standards have come up with safety requirements that are applicable to AI/ML components. It is recommended that any relevant safety requirement/recommendations be mentioned and discussed with the OEM.

4. Overview of Component Development Methodology:

In this section, outline the steps used in the development of the component including verification and validation performed at each step and the overall process of development. For instance, the information may include the following:

- whether the network was custom-built or developed as a combination of pre-existing mature components by extending well-known networks (e.g., ResNet) using transfer learning approaches and the pre-training data set (e.g., ImageNet).
- information on the processes, methods & tools used for the development, validation, maintenance and revisions of this machine learning component as well as general AI/ML components. Additionally, for any tool is used in the development process, any data related to tool qualification and benchmarking of the tool performance is recommended.
- information on the use of standard industry practices and processes for preserving privacy and ensuring non violation PII clause
- Information on DevOps and MLOps strategies employed generally for the development of AI/ML components in the organization and specifically during the development of this specific component. This should throw sufficient light on the processes and tools that are used for maintaining different versions of models, code base and data sets.
- Continuous Improvement (CI) and Continuous Deployment (CD) are becoming prevalent practice in the automotive practice and information related to the readiness and capability of the supplier.

5. Data Set Creation:

The data set development is a crucial activity in building an AI/ML component. There are several steps involved in developing a quality data set:

5.1 Data Set Sourcing: The source of data could be varied and may include experimental, synthetic (GANs, Simulation, Augmentation, etc.) or a mixture. In order to assess the

integrity and reliability of the data, it is important to know the process for data creation.

5.2 Data Set Cleaning: Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. One useful metric providing evidence is the amount of data included/discarded in the process of cleaning.

5.3 Validation: This involves checking the integrity, and appropriateness of the data for the specific applications implemented by the component; These checks involve whether the data set conforms to the expected requirements of the component. For instance, whether the images have been created using sufficient lighting, contrast, brightness, sharpness etc., whether the type and number of objects present in the images are conforming to the task implemented by the component. Another important aspect of validation is completeness of the data set. This involves providing statistical information about the nature, type and number of the various data elements present in the data set. This should enable one to assess whether the dataset is skewed or biased, whether every type of objects and conditions (e.g., lighting, weather) are adequately represented in the population.

5.4 Labeling: Labelling is a key step for supervised machine learning. Industry-standard processes, methods and tools may be followed for labeling the input data set. The details like labeling format, the processes, methods, tools and resources used for labeling shall be provided. This should include whether any automated methods were used in the labeling process and the quality assurance methods used for the validation of annotation.

5.5 Dataset Infrastructure: The creation, development and maintenance of the data set requires careful consideration. There are many recent developments in data set infrastructure, data set version management and the details on the infrastructure platform used for data set maintenance. The details on the infrastructure used in the development of this component are to be included.

Outline the process, methods and tools used involved in the development life cycle of the data set for the model development. This includes

- information and details regarding the data source (synthetic and real),
- cleaning and any other data pre-processing steps employed,
- data augmentation,
- data labeling process
- details regarding evolution and versions of the data set and the associated version controls
- integrity and validation checks performed

For each of the above items, provide appropriate evidence of having performed the steps. This could include information regarding the assets created and appropriate metrics that is a

qualitative or quantitative measure of performance. The samples and statistics of the data set, if not the full data set, used for training and testing are also to be provided.

In Figure 4 below, few examples of training images are shown. The figure contains four images and from left to right, they depict respectively normal traffic, motion blurred image, image with an object occlusion and an augmentation of an image where the car was inserted onto a background image. The training data might also include simulated examples and even adversarial examples.

Figure 4: Sample Training Images

Figure 5 shows examples of dataset distributions according to location, weather, time of day. The distribution shall match the specification of the component in the sense that there is sufficient number of data items in every subset of datasets where the performance of the AI/ML components is expected to be high; poor performance may be acceptable if the performance on an under-represented class of data items is not expected to be high.

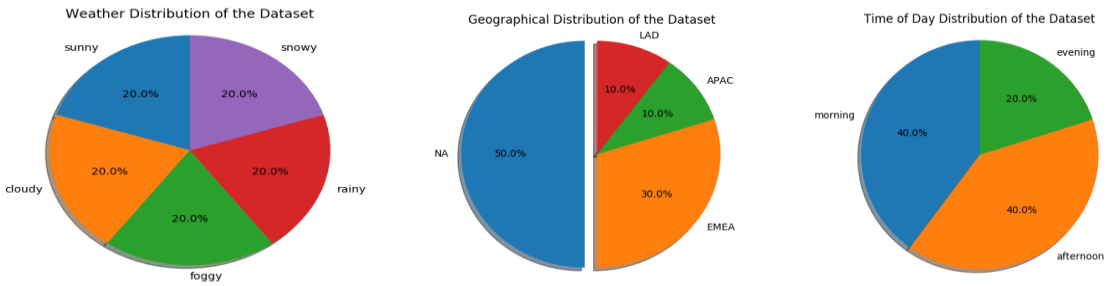


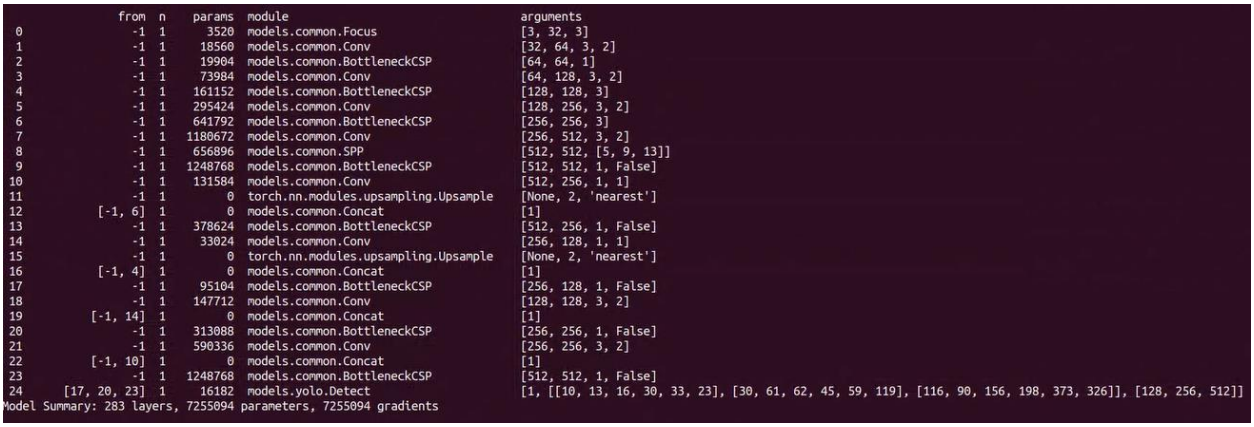
Figure 5: Data Set Distribution

6. **Model Development**:

In this section, furnish details on the model development steps including the model architecture, framework (e.g., Tensor Flow, PyTorch), the tools used during the development, functional performance metrics (e.g., accuracy, recall, IOU) used at different stages of development. The details may include information like whether the model was derived using transfer learning approaches from other models by adding some layers and/or retraining existing layers.

6.1 Model Architecture: Broad details of the model backbones (standardized architectures like CNNs, Encoder-Decoders, Seq-Models etc.), model elements including the type and number of layers, activation functions, size and nature of parameters may be given.

As an example shown in Figure 6, one could give a screenshot from the model development platform that provides information related to the layers and the parameters of a Convolutional Neural Network (CNN).



	from	n	params	module	arguments
0	-1	1	3520	models.common.Focus	[3, 32, 3]
1	-1	1	18560	models.common.Conv	[32, 64, 3, 2]
2	-1	1	19904	models.common.BottleneckCSP	[64, 64, 1]
3	-1	1	73984	models.common.Conv	[64, 128, 3, 2]
4	-1	1	161152	models.common.BottleneckCSP	[128, 128, 3]
5	-1	1	295424	models.common.Conv	[128, 256, 3, 2]
6	-1	1	641792	models.common.BottleneckCSP	[256, 256, 3]
7	-1	1	1180672	models.common.Conv	[256, 512, 3, 2]
8	-1	1	656896	models.common.SPP	[512, 512, [5, 9, 13]]
9	-1	1	1248768	models.common.BottleneckCSP	[512, 512, 1, False]
10	-1	1	131584	models.common.Conv	[512, 256, 1, 1]
11	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
12	[-1, 6]	1	0	models.common.Concat	[1]
13	-1	1	378624	models.common.BottleneckCSP	[512, 256, 1, False]
14	-1	1	33024	models.common.Conv	[256, 128, 1, 1]
15	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
16	[-1, 4]	1	0	models.common.Concat	[1]
17	-1	1	95104	models.common.BottleneckCSP	[256, 128, 1, False]
18	-1	1	147712	models.common.Conv	[128, 128, 3, 2]
19	[-1, 14]	1	0	models.common.Concat	[1]
20	-1	1	313088	models.common.BottleneckCSP	[256, 256, 1, False]
21	-1	1	590336	models.common.Conv	[256, 256, 3, 2]
22	[-1, 10]	1	0	models.common.Concat	[1]
23	-1	1	1248768	models.common.BottleneckCSP	[512, 512, 1, False]
24	[17, 20, 23]	1	16182	models.yolo.Detect	[1, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 119], [116, 90, 156, 198, 373, 326]], [128, 256, 512]]

Model Summary: 283 layers, 7255094 parameters, 7255094 gradients

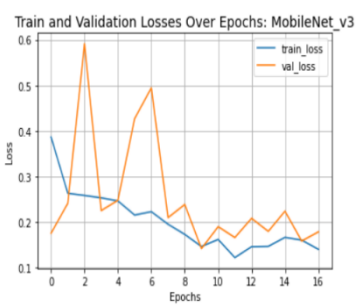
Figure 6: AI Model Architecture Details

6.2 Learning Framework: Machine learning models are typically developed using frameworks like TensorFlow, PyTorch, Keras, etc., and typically use various framework dependent libraries. Details regarding specific libraries and dependencies (versions) by the model may be included.

6.3 Data Set splitting**:** The method used for splitting the data set into training, validation and test data sets may be highlighted. If non-standard data splitting methods are used, then details about these methods must be provided.

6.4 Model Training**:** Relevant information regarding the training may be provided which include: the batch sizes, number of epochs, the hardware platform (e. g., type and number of GPUs used), amount of time, and the model performance metrics (e.g., precision, recall, mAP, etc.) used for measuring the progress and termination of the training. The performance metrics of the final model should also be included.

During training, training and validation losses are computed. The specification could include graphs depicting the losses as given in figure below. Figure 7 gives examples of graphs that reveals that the validation loss and training loss might not agree and whenever they do not, it indicates a possible overfitting.



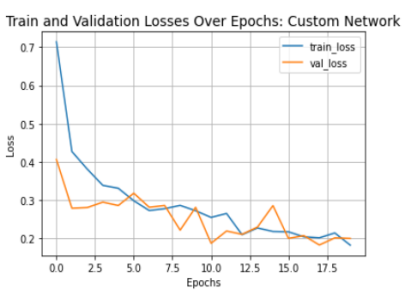
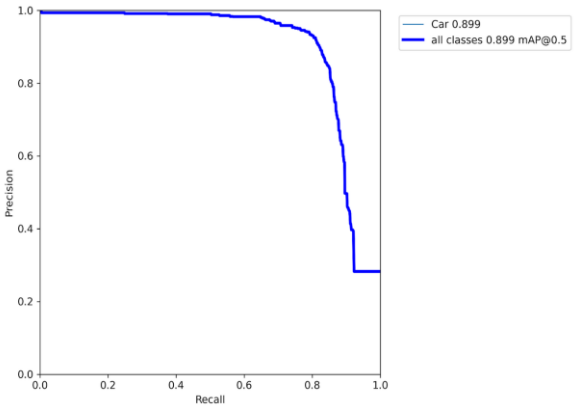


Figure 7: Training and Validation Loss Graphs

6.5 Validation: The purpose of validation is to avoid overfitting by stopping the training of the model at the `right' step. Information related to validation performed during the training stage may be provided. This includes, the number of training epochs after which validation was performed, the results of validation and the evolution of the results over successive validation steps until the final step at which the training was stopped.

Various key performance indicators (KPIs) are used during validation and some popular metrics are precision, recall, F1 and mean Average Precision (mAP) for classification and object detection. These should be traceable to the requirements of the AI function and that of the system using the AI function. The KPIs may include target thresholds for the different metrics and details be given as to how these targets are derived. Figure 8 is an example of mAP where mAP is the area under the recall-precision curve.



$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precison \times recall}{precision + recall}$$

Figure 8: Key Performance Indicators

7. AI/ML Model Verification:

Quantify the performance of the AI/ML component on the test data sets and any additional data sets (blind or double-blind tests) and demonstrate whether and how the component meets the predefined targets set by the OEMs in the V&V requirements. Verification results must include coverage of all inputs as mentioned in the OEM requirements including any additional use cases, edge cases and accepted noise and

distortion levels. All details on the data set including possible samples of verification results and statistical information on the performance of the model during this step must be provided. Some of the standard metrics that is a measure of performance of the component are: confusion matrix, accuracy and recall rates, IoU (Intersection over Union), etc.

8. Target Code Generation:

The AI/ML model is typically developed in a platform that is different from the target platform. Retargeting the model to an implementation platform often involves adjustments to model architecture, parameter values and generating the code that include addition of a number of functions in the target library.

Provide details of the specific target platform and elaborate on the process, methods and tools used in the creation of the embeddable code for deployment; the target platform could be the one specified by the OEM or a better platform and the reasons for the choice may be included. This may include details on any optimizing steps such as pruning, data type conversions etc., and code generators used for arriving at the target code. The target code may include software library functions the details of which also may be provided.

9. Target Code Verification:

The component code that will run on the target platform is obtained after many optimizations, e.g., layer merging, quantization, and code generation and inclusion of new library functions. As a result, its behavior might be different from that of the model. Additional verification needs to be done to ensure that the behavior of the target code does not significantly differ from the model.

In this step, the verification results of the target code and all the supporting library functions are to be provided. The performance of the target code on real inputs, e.g., images from the target camera used in the system, on edge cases and on variations in different sensor parameters like relative sensor placement in the vehicle, viewing angle, height etc. are to be demonstrated. Any performance variation between the target code and the development model may also be quantified.

10. Integrated Feature Verification and Validation (Supplier + OEM):

The AI/ML component is often the part of a feature and its quality and robustness decides the overall quality of the feature. In spite of the verification done in the previous steps, the integrated feature of system might not meet the feature

requirements. This step is to verify the component in its intended context and carry out the verification at the feature level.

In this section, the details of verification and performance results of the system integrated with the target code of the model are to be furnished. This may include the set-up (e.g., HIL Bench, Simulation, Field Testing) used for system validation. A validation report summarizing the performance of the system and the AI/ML component could be included. The summary could elaborate the details of the scenarios encountered by the system and performance metrics related to the tasks carried out by the AI/ML component. The validation step may involve both the OEM and the supplier as the latter supplies only the component and it is the responsibility of the OEM to integrate this with the rest of the system. The performance metrics computed in this step is required to meet the acceptance criteria set forth by the OEM in the requirements.

11. **Post-deployment Component Monitoring and Validation (OEM + Supplier)**:

This step would be performed by the OEM during the post-deployment stage for possible anomalies, performance drifts, new edge cases, etc. that may require additional development and improvement by the supplier. The OEMs may create a performance report either on their own or along with the supplier during the early stages of deployment of the system in the field.
The report may contain edge cases and problematic situations when the component failed to perform as per the intended functionality. The report, for example, could contain images shown in Figure 10.

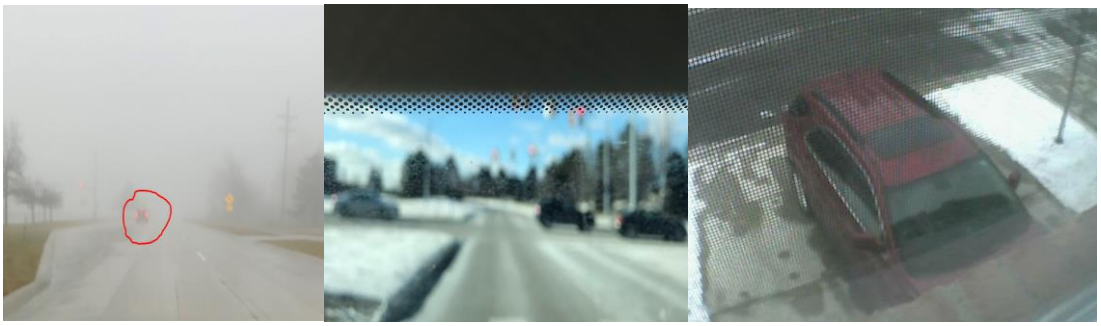


Figure 10: Edge Case Images

This figure contains three images: the left most image was captured in an extreme weather condition which is the outside the ODD and the performance of the component was not satisfactory. The image in the middle is out-of-focus and blurred image and could be an edge

case depending upon the degree of blur. The third one is when the camera is viewing the object from top and this is naturally out of scope and the component need not behave well.

The post deployment validation report may contain examples of such edge cases. The report may contain more comprehensive information on the type of failure cases, their severity and causal factors, e.g., weather conditions, traffic participants, lighting. The report may also highlight how the model and implementation was subsequently refined and improved using those failure cases.

Conclusion

The guidelines specified in the document identifies and enumerates the additional life cycle artefacts to be furnished by the supplier of the AI/ML components. It is believed that these would provide reasonable evidence on the quality and robustness of the component. The guidelines have been designed to be rather lightweight and the reasons for this are many: the state-of-the-art and practice of AI/ML components are at an early stage of maturity, the developers and users need to have more experience in developing or using these components, the guidelines are expected to be complementary and supplants possible guidelines that the OEMs might enforce on their suppliers and the emerging global safety and security standardization efforts may bring in new guidelines.

There is a companion document prepared by the workgroup team to illustrate the application of the guidelines outlined here on a hypothetical example [4]. This would help the suppliers in developing additional artefacts and presenting them to the OEMs.

Acknowledgements: The authors would like to thank Simon Burton, Fraunhofer IKS, Eric Barbier and Prashant Dhurjati, Wayve Technologies Ltd., and several other colleagues from the three OEMs and who provided very useful comments and feedback on the earlier drafts

Bibliography

1. A. Karpathy, Software 2.0, https://karpathy.medium.com/software-2-0-a64152b37c35

2. Automotive SPICE Process Reference Model, Ver. 3.0, https://www.automotivespice.com/fileadmin/software-download/Automotive_SPICE_PAM_30.pdf

3. Safety and Artificial Intelligence Committee, TC 22 SC 32 WG 14

4. DL-SPICE Specification (Sample), USCAR Document, 2021.

5. https://wayve.ai/company