

# Out-of-Distribution Detection as Support for Autonomous Driving Safety Lifecycle

Jens Henriksson<sup>1</sup>, Stig Ursing<sup>1</sup>, Murat Erdogan<sup>2</sup>,  
Fredrik Warg<sup>3,[0000-0003-4069-6252]</sup>, Anders Thorsén<sup>3,[0000-0001-7933-3729]</sup>,  
Johan Jaxing<sup>4</sup>, Ola Örsmark<sup>5</sup> and Mathias Örtenberg Toftås<sup>1</sup>

<sup>1</sup> Semcon, dept. Software and Emerging Tech, Gothenburg, Sweden  
{jens.henriksson, stig.ursing, mathias.ortenberg-toftas}@semcon.com

<sup>2</sup> Veoneer, Linköping, Sweden, murat.erdogan@veoneer.com

<sup>3</sup> RISE Research Institutes of Sweden, Borås, Sweden  
{fredrik.warg, anders.thorsen}@ri.se

<sup>4</sup> Agreat, Gothenburg, Sweden, johan.jaxing@agreat.com

<sup>5</sup> Comentor, Gothenburg, Sweden, ola.orsmark@comentor.se

This version of the contribution has been accepted for publication after peer review but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: [10.1007/978-3-031-29786-1\\_16](https://doi.org/10.1007/978-3-031-29786-1_16). Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use <https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms>.

# Out-of-Distribution Detection as Support for Autonomous Driving Safety Lifecycle\*

Jens Henriksson<sup>1</sup>, Stig Ursing<sup>1</sup>, Murat Erdogan<sup>2</sup>,  
Fredrik Warg<sup>3</sup>, [0000–0003–4069–6252], Anders Thorsén<sup>3</sup>, [0000–0001–7933–3729] ✉,  
Johan Jaxing<sup>4</sup>, Ola Örsmark<sup>5</sup> and Mathias Örtenberg Toftås<sup>1</sup>

<sup>1</sup> Semcon, dept. Software and Emerging Tech, Gothenburg, Sweden  
{jens.henriksson, stig.ursing, mathias.ortenberg-toftas}@semcon.com

<sup>2</sup> Veoneer, Linköping, Sweden, murat.erdogan@veoneer.com

<sup>3</sup> RISE Research Institutes of Sweden, Borås, Sweden  
{fredrik.warg, anders.thorsen}@ri.se

<sup>4</sup> Agreat, Gothenburg, Sweden, johan.jaxing@agreat.com

<sup>5</sup> Comentor, Gothenburg, Sweden, ola.orsmark@comentor.se

**Abstract.** [Context and Motivation] The automotive industry is moving towards increased automation, where features such as automated driving systems typically include machine learning (ML), e.g. in the perception system. [Question/Problem] Ensuring safety for systems partly relying on ML is challenging. Different approaches and frameworks have been proposed, typically where the developer must define quantitative and/or qualitative acceptance criteria, and ensure the criteria are fulfilled using different methods to improve e.g., design, robustness and error detection. However, there is still a knowledge gap between quality methods and metrics employed in the ML domain and how such methods can contribute to satisfying the vehicle level safety requirements. [Principal Ideas/Results] In this paper, we argue the need for connecting available ML quality methods and metrics to the safety lifecycle and explicitly show their contribution to safety. In particular, we analyse Out-of-Distribution (OoD) detection, e.g., the frequency of novelty detection, and show its potential for multiple safety-related purposes. I.e., as (a) an acceptance criterion contributing to the decision if the software fulfills the safety requirements and hence is ready-for-release, (b) in operational design domain selection and expansion by including novelty samples into the training/development loop, and (c) as a runtime measure, e.g., if there is a sequence of novel samples, the vehicle should consider reaching a minimal risk condition. [Contribution] This paper describes the possibility to use OoD detection as a safety measure, and the potential contributions in different stages of the safety lifecycle.

**Keywords:** Automotive safety · Out-of-Distribution detection · Machine learning · Automated driving systems · Safety requirements

---

\* This research has been supported by the Strategic vehicle research and innovation (FFI) programme in Sweden, via the project SALIENCE4CAV (ref. 2020-02946) and by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation.

## 1 Introduction

Machine learning (ML) techniques are increasingly used in many domains to solve problems where rule-based algorithms are difficult or impractical to construct or do not scale. One common use is for object detection and classification in images/video, an area where large advances have been made in the last decade. This is often used for perception systems in robotics as a way to create a world model the machine can use to make decisions on future actions. In some cases, ML is used not only for environment perception but also for decision making. A challenge is determining how well ML will perform in a given application, especially for open-world problems with a virtually infinite variation of environmental conditions, edge cases and potentially adversarial attacks [18].

One such application is automated driving systems (ADS) [15], which are seen as a key technology for more efficient, available and safe mobility. However, for such safety-critical systems, the challenge to determine ML performance also becomes a safety issue. While there are several frameworks and standards, see Sec. 2, for ML in safety-critical applications, and many methods available for improving specification, testing and robustness, or performing run-time error detection, we argue that there is still a gap when it comes to analysing exactly how the available methods can contribute to fulfilling the safety requirements.

As a step towards a better understanding of how to combine the available methods and metrics for ML performance with the need for safety assurance, this paper analyses Out-of-Distribution (OoD) detection in light of a safety lifecycle. The general concept of OoD detection is to distinguish known objects/samples from unknown, e.g., detecting if an input to the ML model is similar enough such that the model may provide an accurate prediction, something that is less likely for a sample of an unknown distribution. Based on this analysis, we identified the following potential roles for OoD detection in a safety lifecycle:

- **Development phase:** During development it may be used to identify limitations in the training dataset through highlighting scenarios where the detection rate is below the required pass/fail threshold. Alternatively, it can suggest operational design domain (ODD) [15] reduction if certain scenarios are not fulfilling the allocated safety requirements.
- **Shadow mode:** From a continuous experimentation and deployment perspective, it could be used to test the expansion of ODD boundaries and highlight scenarios where more training data is needed.
- **Operational phase:** During run-time, OoD detection may help to identify uncertainties in the deep neural network (DNN) and trigger safe fallback routines if the uncertainty in the model goes above pre-defined threshold.

The remainder of this paper connects OoD detection to state-of-the-art safety-related research, followed by a description on how to incorporate OoD detection as a safety measure for different stages of the lifecycle of ML models in an ADS. Our vision is that this work will inspire more experimental work that demonstrates the effect on safety for various ML improvement techniques.

## 2 Related work

The functional safety standard *ISO 26262* [10] is essential in automotive development. It deals with hazards caused by technical failures due to random and systematic faults in the system’s hardware or software. However, the existing version does not fully cover safety requirements of ADS features relying on environment perception or ML [17]. A major issue with ML based systems is that they are not fully specifiable, while ISO 26262 implicitly assumes that all functionality is specified [17, 16, 8].

Increased automation raised the need to complement ISO 26262 with the safety of the intended functionality standard *ISO 21448 (SOTIF)* [12] dealing with hazards due to functional insufficiencies in the absence of system failures. Potential hazardous behaviour includes the inability to correctly perceive the environment, the lack of robustness with respect to sensor input variations, and unexpected behaviour by the decision making algorithm. All factors relevant for ML based solutions. The product development activities specified in ISO 21448 can be carried out in parallel with the activities in ISO 26262 as illustrated in Fig. 1. Other relevant standardization works in progress related to safety assessment of road vehicles relying on ML include *ISO/IEC TR 24029*, *ISO/IEC AWI TR 5469*, and *ISO/AWI PAS 8800*.

Burton et al. presented a work about safety assurance of ML for perception functions including an analysis of ISO 26262 and ISO 21448 [3]. The authors argue that due to the typical failure modes and performance limitations of ML, an absolute level of correctness is infeasible. Instead, in line with ISO 21448, quantitative assurance targets are required defining an acceptable limit to the probability that guarantees cannot be met. Mohseni et al. presented an extensive review and propose a taxonomy of ML safety that maps state-of-the-art ML techniques to key safety engineering strategies [13]. Due to the lack of verification techniques for deep neural networks, ML model validation commonly relies on accuracy measurements on different large tests sets to cover the targeted ODD. This is an important metric of the success of the algorithm, but it fails to capture

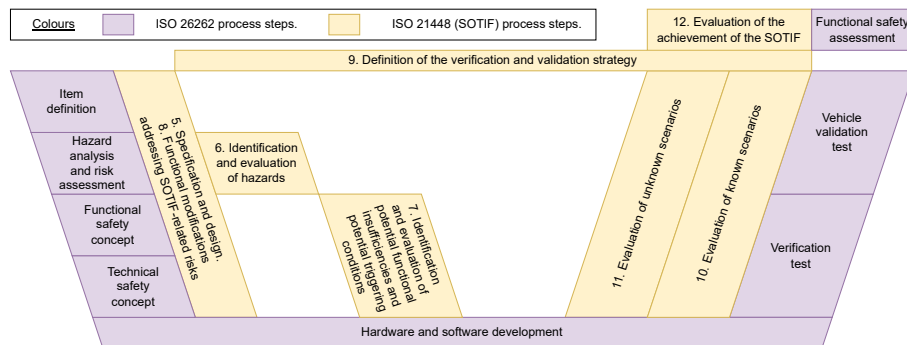


Fig. 1. Possible combined ISO 26262 and ISO 21448 development cycle (based on [12]).

the model’s confidence in its predictions. For example, if an image classifier trained to identify pedestrians and road signs was presented with a billboard showcasing a person, and the model had not encountered such examples during training, it could output class probabilities from anywhere between 100% human to 100% road sign. What these probabilities fail to show is the model’s lacking confidence.

Borg et al. [1] studies an autonomous emergency breaking (AEB) system constructed in a simulator. The approach incorporates a *safety-cage* implemented through a variational autoencoder to detect samples that are OoD, and uses the simulator to do rigorous testing such that the system passes safety requirements. The requirements are assigned to the ML model that operates on the front looking camera. The same approach is used in this paper.

OoD detection is one way of dealing with the fact that data used to train an ML model rarely covers all possible scenarios the model will face when put into production. OoD deals with this open world assumption by dividing data into In-Distribution (ID) and OoD, where ID is typically data close to what was seen during training and OoD is unrecognized data. In the article [19] Yang et al. presents a unified framework for generalized OoD detection. This framework will be used as a base for discussing the potential of OoD detection in the ML-lifecycle.

Yang et al. [19] divides the larger subject of general OoD detection into sub-topics, namely: Anomaly detection, Novelty detection, Open set recognition/OoD detection, and outlier detection. The split is motivated with four dichotomies: whether covariate or semantic shift is detected; whether the ID set is treated as a single or multiple classes; whether ID classification is required; and whether the method uses inductive or transductive reasoning. Covariate and semantic shift refers to a difference between the data distributions of the training and testing sets, the two shifts differ in whether the distribution change occurs in the input data or the labels respectively. Inductive reasoning makes use of the entire dataset to make specific inferences on said data, this differs from transductive reasoning that attempts to learn general rules from a training dataset which is later applied to a test dataset.

The binary classification between ID and OoD, i.e., Anomaly and Novelty Detection, typically treats the entire ID as one class [19]. One of its applications is detecting anomalies in data streams and can be used for data mining [19, 14]. An unexplored area within this concept is monitoring data streams from vehicles to identify uncommon traffic scenes.

Open Set Recognition (OSR) and OoD detection are closely related to each other and differentiate themselves from the other approaches by not limiting the detection to binary classification of ID or OoD, but instead allowing multi-class classification of ID samples while still classifying OoD samples as such [19]. OSR and OoD detection is not limited to simple image classification, examples of other areas where it has been applied is object detection [7, 5], image segmentation [4], and 3D object detection [9].

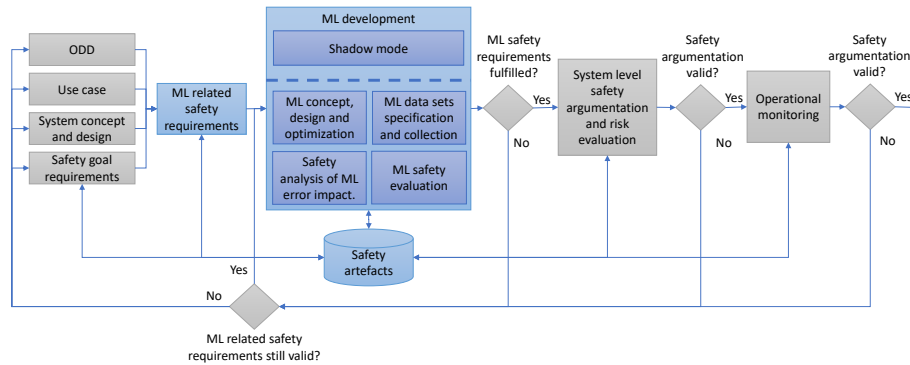


Fig. 2. View of lifecycle for ML based system (inspired by [12, 11, 3]).

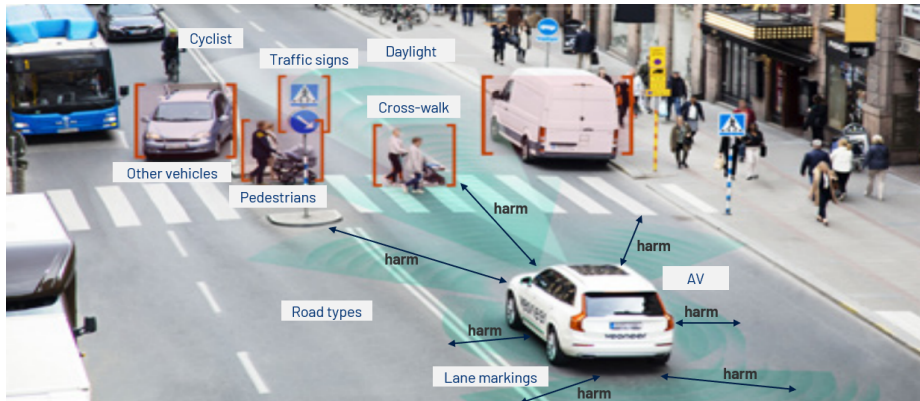
### 3 Methodology

A possible methodology covering the safety lifecycle for an ML based system is shown in Fig. 2. The lifecycle is inspired by the work of Burton et al. [3] and is compatible with the ISO 26262 and ISO 21448 development cycle shown in Fig. 1. On the left side system level requirements are shown including safety goal requirements. In the next step these are broken down to ML related safety requirements that are input to the ML development phase.

A safety lifecycle typically starts by deriving safety goals, which are the top-level safety requirements, using hazard analysis and risk assessment. These safety goals are addressed through refined safety requirements introduced at different stages of the lifecycle. Fig. 2 visualizes how the safety lifecycle can be extended for ML components using an iterative process that aims to extract safety artefacts and refine safety requirements throughout the development process to ensure correct behaviour during operation.

An issue with safety goals is the abstraction level. As they will typically express safe behaviour on vehicle level, refinement to safety requirements for different subsystems of the ADS will be necessary. Our suggestion is to construct a system abstraction to aid the refinement of relevant safety requirements to the respective systems. The abstraction is further elaborated in Sec. 3.1.

Incorporating ML into ADS has been summarized as challenging due to the inherent lack of understanding the ML behavior. Safety requirements can still be applied to systems that incorporate ML, as described in SOTIF. However, determining which methods, such as pass or fail criteria, to use in order to show compliance with the safety requirements are still lacking, both from academia and industry. We propose in this paper the OoD detection method as one potential criteria to determine the compliance level of ML safety requirements. We describe how this technique can be used in the development stage of the lifecycle in Sec. 3.2 and in the verification and operational stages of the lifecycle in Sec. 3.3.



**Fig. 3.** Photo of an AEB equipped vehicle approaching a cross walk, illustrating a scene in the use case. Brackets indicate objects that must be included in the ODD. Arrows are used to illustrate possible sources of harm.

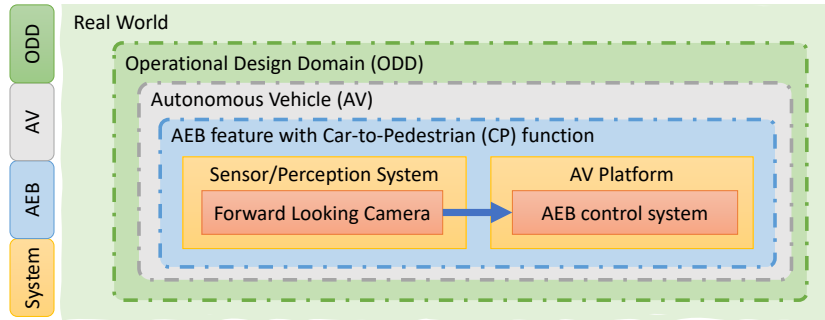
### 3.1 Autonomous Emergency Breaking Use-Case

As a relatively simple example to demonstrate the ML safety lifecycle we use an Autonomous Emergency Breaking (AEB) system. Among other goals, this system aims to avoid collisions with pedestrians by performing emergency breaks when said collisions are predicted. A scene from such a use case is shown in Fig. 3 with brackets indicating some of the objects that must be included in the ODD description for the AEB. Note that a complete ODD description must include much more information [6].

As mentioned, common practice in the automotive domain to define the high level safety goals and a safety concept is to follow the process shown in Fig. 1, starting with an item definition followed by a hazard analysis and risk assessment. For the AEB example, one safety goal resulting from this analysis could be to avoid harm to pedestrians. This is illustrated in Fig. 3 with arrows and the text 'harm' pointing to objects critical to avoid.

For a systematical breakdown of the top level safety goal to concrete safety requirements we propose to define the abstraction levels visualized in Fig. 4. The abstraction levels are defined as:

- ODD: The operational design domain defined as a set of operating conditions. Some relevant conditions are visualized in Fig. 3.
- AV: Autonomous Road Vehicle equipped with an AEB feature.
- AEB: The autonomous emergency braking system.
- Sensor/Perception System: Forward Looking Camera containing the DNN based ML algorithm.



**Fig. 4.** A figure showing our proposed abstraction levels, in this case specified to our AEB use-case.

The high-level safety requirements are refined and allocated to the different systems in Fig. 4 after a fault tree analysis and system decomposition. We analyzed and designed the forward looking camera in the 'Sensor/Perception Platform' to satisfy the allocated safety requirements. For that purpose we developed a technical safety concept and derived technical safety requirements towards the perception model within the forward looking camera, which uses a DNN algorithm for object detection. Here we have used and modified safety requirements allocated to machine learning from [1] alongside the lifecycle in Fig. 2.

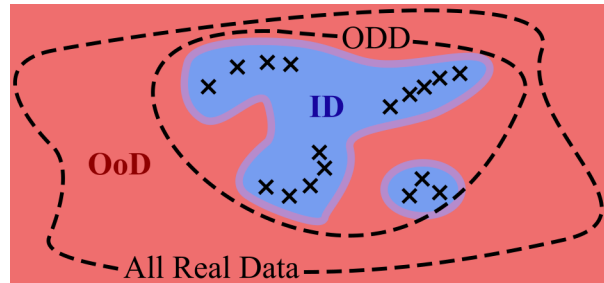
Below are some examples from the derived ML safety-related requirements that are allocated to the ML component [1]:

- SYS1-FLC-PER-REQ 1: The false negative rate of the perception algorithm within the forward looking camera shall not exceed 7% within 50ms.
- SYS1-FLC-PER-REQ 2: The false positive per image of the perception algorithm within the forward looking camera shall not exceed 0.1% within 80ms.

### 3.2 Machine Learning Development

With the initial safety requirements for the ML component in place, the ML development block in Fig. 2 can be pursued. Within the model optimization block, the process for training the model is defined, which encompasses the labeling format to fulfill the functional requirements, the model architecture type, and the optimization methods (such as the optimization strategy, training length, and loss evaluation). The block also marks the first use of OoD detection to identify underrepresented areas in the ODD and set the threshold for classifying data as ID or OoD.

If the training results are insufficient, two options are available: either improving the model performance or adjusting the functional scope. To enhance the model, acquiring additional training data or applying better architectures,



**Fig. 5.** The distribution of the models input data. The crosses mark discrete data points, the blue region corresponds ID regions, and the red region represents the OoD region.

techniques, or mitigation strategies may help. However, if the performance still falls short, adjusting the functional scope through refining the requirements or reducing the size of the ODD, particularly in weakly covered areas identified by the OoD detection, becomes necessary. Fig. 5 offers a visual representation of the relationship between the ODD, the OoD, and the ID. In the figure, crosses denote discrete data points in the input data space, while the blue and red regions indicate areas that the model has comprehended (ID) or failed to (OoD). For the depicted case, the model has not covered all regions inside the ODD, therefore, the ODD must either be reduced or more data must be collected.

To determine how ML errors may impact the safety, we followed the safety analysis of Burton et al. [3]. Some of their conclusions can be seen in Fig. 6, these helped inform us how to formulate our safety requirements. One of the main causes of these safety errors lies in distributional shift, which can be mitigated using OoD as described in this section.

### 3.3 Remaining Lifecycle

Once the AEB System that incorporates ML has demonstrated acceptable performance during the ML development phase, the system level integration and verification and validation process can begin. During this step, the system level performance is evaluated to confirm that it complies with the previously generated safety requirements, thus ensuring that the system level safety argumentation is valid.

As part of verifying that the safety requirements are fulfilled, one can use shadow mode operation, i.e., the feature is deployed in the field but not allowed to control the vehicle. Instead the feature continuously collects and categorizes data into ID or OoD in the background. This allows for the identification of scenarios that are still challenging for the model, which in turn provides insights into how the current ML component can be improved through more focused data collection or ODD modification.

Subsequently, the AV may be deployed. It is at this point that operational monitoring commences, through the utilization of OoD detection to identify

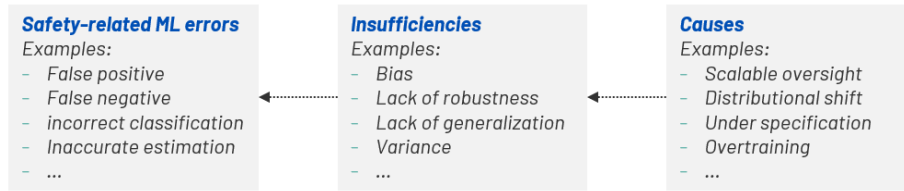


Fig. 6. Fault-model of safety-related ML errors (based on [2]).

scenarios where the system is operating in uncertain conditions, such as those that were not well represented during the training phase and may not have been generalized appropriately or at the edge of the ODD. If such scenarios arise, the system may revert to a safe state, such as achieving a minimal risk condition (where the vehicle is at a standstill) or transferring control to a human driver.

## 4 Conclusion and Outlook

This paper has introduced system abstraction levels for an ADS that allows for a breakdown of safety goals into safety requirements that can directly be allocated to the necessary systems, especially with a focus on how to derive safety requirements for subsystems making use of ML. This allows for separation of the existing concerns in the system, e.g., faults, errors or insufficiencies in the functionality.

OoD detection is one technique that is beneficial to validate the system to be within acceptable error margins. The technique allows for risk mitigation in ML components as it aims to reduce the false positive and negative samples. It is shown how this method can be used for varying purposes at different stages of the safety lifecycle.

Going forward, more experiments are needed on safety assurance techniques for ML to establish the actual performance of the system. For instance, while we have outlined how one can use OoD in the safety lifecycle, the extent of the contribution from this technique towards fulfilling the safety requirements is still unknown and will need more extensive experiments. An aim with this work is also to highlight the need to combine runtime evaluation techniques to combat the uncertainty that exist within the ML component.

Our vision and hope is to inspire more experimental evaluation and analysis into how OoD and other methods from the ML domain can be applied in a safety lifecycle, in particular how each method can help fulfill safety requirements allocated to ML components. Such knowledge will be crucial for determining how and when ML can be used in the design of safety-critical systems.

## References

1. Borg, M., Henriksson, J., Socha, K., Lennartsson, O., Lönegren, E.S., Bui, T., Tomaszewski, P., Sathyamoorthy, S.R., Brink, S., Moghadam, M.H.: Ergo, smirk is safe: A safety case for a machine learning component in a pedestrian automatic emergency brake system. arXiv preprint arXiv:2204.07874 (2022)
2. Burton, S.: A causal model of safety assurance for machine learning. arXiv preprint arXiv:2201.05451 (2022). <https://doi.org/10.48550/arXiv.2201.05451>
3. Burton, S., Hellert, C., Hüger, F., Mock, M., Rohatschek, A.: Safety Assurance of Machine Learning for Perception Functions. In: Deep Neural Networks and Data for Automated Driving: Robustness, Uncertainty Quantification, and Insights Towards Safety, pp. 335–358. Springer International Publishing (2022)
4. Cen, J., Yun, P., Cai, J., Wang, M.Y., Liu, M.: Deep metric learning for open world semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15333–15342 (2021)
5. Du, X., Wang, X., Gozum, G., Li, Y.: Unknown-aware object detection: Learning what you don’t know from videos in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13678–13688 (2022)
6. Gyllenhammar, M., Johansson, R., Warg, F., Chen, D., Heyn, H.M., Sanfridson, M., Söderberg, J., Thorsén, A., Ursing, S.: Towards an operational design domain that supports the safety argumentation of an automated driving system. In: Proceedings of ERTS 2020. Toulouse, France (Jan 2020)
7. Hendrycks, D., Basart, S., Mazeika, M., Mostajabi, M., Steinhardt, J., Song, D.: Scaling out-of-distribution detection for real-world settings. arXiv preprint arXiv:1911.11132 (2019)
8. Hoss, M., Scholtes, M., Eckstein, L.: A Review of Testing Object-Based Environment Perception for Safe Automated Driving. *Automotive Innovation* **5**(3), 223–250 (Aug 2022). <https://doi.org/10.1007/s42154-021-00172-y>
9. Huang, C., Nguyen, V.D., Abdelzad, V., Mannes, C.G., Rowe, L., Therien, B., Salay, R., Czarnecki, K.: Out-of-distribution detection for lidar-based 3d object detection. arXiv preprint arXiv:2209.14435 (2022)
10. ISO: 26262:2018 Road Vehicles – Functional Safety. ISO (2018)
11. ISO: ISO/TR 4804:2020 Road Vehicles - Safety and Cybersecurity for Automated Driving Systems - Design, Verification and Validation. ISO (2020)
12. ISO: 21448:2022 Road Vehicles – Safety of the Intended Functionality. ISO (2022)
13. Mohseni, S., Wang, H., Yu, Z., Xiao, C., Wang, Z., Yadawa, J.: Taxonomy of Machine Learning Safety: A Survey and Primer. arXiv:2106.04823 [cs] (Mar 2022)
14. Ramachandra, B., Jones, M., Vatsavai, R.R.: A survey of single-scene video anomaly detection. *IEEE trans. on pattern analysis and machine intelligence* (2020)
15. SAE: J3016 Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. Tech. Rep. J3016:2021, SAE Int. (Apr 2021)
16. Salay, R., Czarnecki, K.: Using Machine Learning Safely in Automotive Software: An Assessment and Adaption of Software Process Requirements in ISO 26262. arXiv:1808.01614 [cs, stat] (Aug 2018)
17. Salay, R., Queiroz, R., Czarnecki, K.: An Analysis of ISO 26262: Using Machine Learning Safely in Automotive Software. Arxiv preprint 1709.02435. (2017)
18. Tencent Keen Security Lab: Experimental Security Research of Tesla Autopilot. Tech. rep., (Mar 2019), [https://keenlab.tencent.com/en/whitepapers/Experimental\\_Security\\_Research\\_of\\_Tesla\\_Autopilot.pdf](https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf)
19. Yang, J., Zhou, K., Li, Y., Liu, Z.: Generalized out-of-distribution detection: A survey. arXiv preprint arXiv:2110.11334 (2021)