Towards Safety Assurance of Uncertainty-Aware Reinforcement Learning Agents

Felippe Schmoeller Roza[1], Simon Hadwiger[2,3], Ingo Thorn[2] and Karsten Roscher[1]

[1]Fraunhofer IKS, Munich, Germany
[2]Siemens AG, Nuremberg, Germany
[3]University of Wuppertal, Wuppertal, Germany

Abstract
The necessity of demonstrating that Machine Learning (ML) systems can be safe escalates with the ever-increasing expectation of deploying such systems to solve real-world tasks. While recent advancements in Deep Learning reignited the conviction that ML can perform at the human level of reasoning, the dimensionality and complexity added by Deep Neural Networks pose a challenge to using classical safety verification methods. While some progress has been made towards making verification and validation possible in the supervised learning landscape, works focusing on sequential decision-making tasks are still sparse. A particularly popular approach consists of building uncertainty-aware models, able to identify situations where their predictions might be unreliable. In this paper, we provide evidence obtained in simulation to support that uncertainty estimation can also help to identify scenarios where Reinforcement Learning (RL) agents can cause accidents when facing obstacles semantically different from the ones experienced while learning, focusing on industrial-grade applications. We also discuss the aspects we consider necessary for building a safety assurance case for uncertainty-aware RL models.

Keywords
Uncertainty estimation, Distributional shifts, Reinforcement Learning, Functional Safety

1. Introduction

This position paper is presented to serve as motivation for the long-term objective of using the uncertainty estimation capabilities of a Reinforcement Learning (RL) agent to improve its functional safety and enable RL as a viable framework to be deployed in industrial-grade applications. Although not a new concept, recent accomplishments have reignited the interest in using RL as a viable method to obtain agents able to interact with a wide range of environments (see [1, 2, 3]). These results were only possible due to the integration of Deep Neural Networks (DNNs) as function approximators for RL agents.

According to some authors (e.g., [4, 5, 6]), the industry is eager to apply Machine Learning (ML) and DNNs more broadly in their processes, with the possibility to increase the safety level by aiding humans in processes that are potentially harmful or even automate complex tasks beyond human capabilities. According to [7], possible applications include aircraft control, power systems, medical systems, and the automotive domain. However, despite the expected gains, industrial players are historically very conservative and, most of the time, only adopt new technologies when there is enough evidence supporting their reliability and cost-effectiveness, which is still not possible for some ML paradigms.

DNNs excel at learning complex representations from a bulk of data, allowing to reach state-of-the-art performance in tasks such as computer vision, natural language processing, and control of autonomous systems. However, DNNs are too complex and have too many parameters to be verified using standard verification and validation methods. On top of that, DNN models are often overconfident and incapable of recognizing that their predictions might be wrong [8]. The combination of these factors has put DNNs at the center of safe AI research in the past few years. The main goal is to guarantee that DNNs can be safe, reliable, secure, robust, explainable, and fair [7].

Another difficulty with DNNs, which also extends to Deep RL, is formalizing how capable they are of generalizing over novel instances. Despite the excellent results obtained with known benchmarks, different findings show that DNNs are susceptible to distributional shifts (e.g., [9, 10]). That means that the model output is not reliable when fed with data drawn from a distribution that differs from its training data distribution, i.e., out-of-distribution (OOD) instances. When considering autonomous systems controlled by RL agents, there is the risk of accidents when facing OOD scenarios. This issue can be solved by making sure the model is trained with data that covers every aspect it might encounter after deployment, which is intractable for open-world complex tasks. Alternatively, some methods have been suggested to make DNNs robust to distributional shifts, such as in [11]. However, making DNNs able to handle

SafeAI 2023: The AAAI's Workshop on Artificial Intelligence Safety, Feb 13-14, 2023 | Washington, D.C., US
felippe.schmoeller.da.roza@iks.fraunhofer.de (F. S. Roza)

distributional shifts is a challenging task and the existing methods are limited. We follow a different direction, which consists in using a monitor to identify the OOD instances. Once OOD is detected, the system can switch to a safe control policy to avoid accidents caused by the agent's inabilities (that could be as simple as "stop and wait for help"). We follow the hypothesis that uncertainty should grow higher when facing the unknown (same as given in [12]) and use uncertainty estimation as a proxy metric to classify OOD inputs.

1.1. Scope and structure of the paper

This paper aims at showing how uncertainty-based OOD detection can help in the long-term goal of building a solid safety case for RL agents, which must be backed by convincing safety arguments. That is not the only factor necessary to make certification of RL models possible, but one of the most important aspects. The paper will focus on industrial applications of automated guided vehicles (AGVs). Industrial environments are mostly guided by specific regulations that are helpful when outlining the system requirements and specifications in terms of safety. We believe this can also be used as a starting point when expanding the framework to a more general case, covering a larger range of open-world applications.

To validate the potential of this approach to help with deriving strong safety arguments, experiments with an environment that simulates the application of transporting goods with a vision-based AGV in warehouses were conducted. The obtained results indicate that uncertainty estimation and OOD detection can help to identify unknown situations which, in some cases, lead to accidents. At the end of the document,

The document is structured as follows: section 2 shows publications available in the literature to serve as background and motivation for this paper. In section 3 the uncertainty-aware RL algorithm is shown. Section 4 contains the experiments and preliminary results, and section 5 presents a short discussion and the future steps we believe are necessary for building the safety assurance case for uncertainty-aware RL systems.

2. Related Work

Publications investigating safety assurance cases for RL systems are limited. Therefore, we will start with relevant works that cover the application of general AI methods in safety-critical applications. That will be followed by works that deal with uncertainty estimation and OOD detection for ML systems, mainly focusing on computer vision problems, and finally, publications that combine uncertainty and RL will be shown. Our work is an intersection of those three topics, with the proposed method being inspired by existing uncertainty quantification approaches and the future outline borrowing ideas from authors that intend to conform AI systems to safety certification processes that are, to the best of our knowledge, very limited when it comes to RL.

AI for safety-critical applications: Different authors defend that to enable ML models to solve safety-critical tasks, the models must be assured by evidence that the ML components will behave in accordance with existing safety specifications. [13] argue that the evidence must cover all aspects necessary to show why these components can be trusted. The authors also present a survey with different methods that help in collecting the evidence for the whole ML lifecycle. In [7], an extensive study in neural networks applied to high assurance systems is presented. In [14], the authors identify problems that arise when using ML following ISO 26262, a standard that regulates the functional safety of road vehicles. They claim that the use of ML can result in hazards not experienced with conventional software. [15] also discuss the shortcomings of fitting ML systems to ISO 26262 and how the Safety of the Intended Functionality (SOTIF), published in the ISO PAS 21448, offers a better alternative for safety assurance. The authors also present an extensive list of safety concerns related to DNN models, including the risk of the data distribution not being a good approximation of the real world and the possibility of distributional shifts to happen over time. [16] also argue that the analysis of ML systems is fundamentally incompatible with traditional safety verification since safety engineering approaches focus on faults at the component level and their interactions with other system components while systemic failures experienced in complex systems are not necessarily consequence of faults from individual parts of the system. Therefore, the safety arguments should also reflect the inherent complexity and unpredictability of ever-changing environments where ML systems are designed to operate.

Machine Learning and Uncertainty: The impact of uncertainty in Machine Learning is a recurrent topic of research, with a plentiful of publications discussing how ML systems should manage uncertainty and presenting methods to quantify uncertainty. In [17], the authors present a more general discussion on the properties of Bayesian Deep Learning models used for computer vision tasks that are affected by aleatoric and epistemic uncertainties (the first is inherent to the system stochastic properties while the former is related to a lack of knowledge). In [18], an introduction to the topic of uncertainty in ML models is provided as well as an overview of the main methods for capturing and handling uncertainty. In [19], the authors show how autonomous systems are affected

by uncertainty and how correctly assessing uncertainty can help towards improving the supervision of inherently unsafe AI systems. Furthermore, a conceptual framework for dynamic dependability management based on uncertainty quantification is presented. In [20], uncertainty quantification as a proxy for the detection of OOD samples is discussed, with different methods compared in image classification datasets, namely CIFAR-10, GTSRB, and NWPU-RESISC45. Some popular uncertainty quantification methods for DNN models worth of mentioning are Monte Carlo Dropout [21], Deep Ensembles [22], and Evidential Deep Learning [23].

Reinforcement Learning and Uncertainty: Most of the work combining uncertainty quantification and ML cover Supervised Learning, with a strong focus on computer vision tasks. However, some literature also shows how uncertainty-aware RL agents can be obtained. A popular application is to use uncertainty to improve exploration. This class of algorithms is motivated by the principle of Optimism in the Face of Uncertainty (OFU) and describes the tradeoff between using high-confidence decisions, that come from the already established knowledge, and the agent's need to explore state-action pairs with high epistemic uncertainty [24].

However, this paper will rather focus on uncertainty as a proxy for detecting domain shifts in decision-making agents. In [25] it is proposed to define the data distributions in terms of the elements that compose a Markov Decision Process (MDP), where minor disturbances should fall under the generalization umbrella and large deviations represent OOD samples. However, determining which semantic properties represent such changes and how to measure them is left as an open question. In [26], the authors present an uncertainty-aware model-based learning algorithm that adds statistical uncertainty estimates combining bootstrapped neural networks and Monte Carlo Dropout to its collision predictor. Mobile robot environments are used to show that the agent acts more cautiously when facing unfamiliar scenarios and increases the robot's velocity when it has high confidence. In [27] this method is extended to environments with moving obstacles. The authors also combine Monte Carlo dropout and deep ensembles with LSTM models to obtain uncertainty estimates. A Model Predictive Controller (MPC) is responsible to find the optimal action that minimizes the mean and variance of the collision predictions.

3. Background

In this section, we present the background for each component of the proposed uncertainty-aware RL algorithm. Different uncertainty quantification methods could be used, but Variational Auto Encoders (VAEs) are an interesting choice for vision-based systems. They are considered robust models, are trained in an unsupervised manner (i.e., labeling samples is not necessary), are fast to train, and their generalization capabilities can be visually inspected by comparing the input and reconstructed images. However, the safety argumentation would benefit from a comparison between different alternatives, with the strengths and deficiencies of each approach addressed, which will remain as a future work suggestion.

3.1. Reinforcement Learning

In RL, we consider an agent that sequentially interacts with an environment modeled as an MDP. An MDP is a tuple $\mathcal{M} := (S, A, R, P, \mu_0)$, where S is the set of states, A is the set of actions, $R : S \times A \times S \mapsto \mathbb{R}$ is the reward function, $P : S \times A \times S \mapsto [0, 1]$ is the transition probability function which describes the system dynamics, where $P(s_{t+1}|s_t, a_t)$ is the probability of transitioning to state s_{t+1}, given that the previous state was s_t and the agent took action a_t, and $\mu_0 : S \mapsto [0, 1]$ is the starting state distribution. At each time step, the agent observes the current state $s_t \in S$, takes an action $a_t \in A$, transitions to the next state s_{t+1} drawn from the distribution $P(s_t, a_t)$, and receives a reward $R(s_t, a_t, s_{t+1})$.

3.2. Variational Auto Encoders

VAEs are a popular class of deep probabilistic generative models [28]. Autoencoders follow a simple encoder-decoder structure, where the model parameters are optimized to minimize the difference between the input sample and the decoded data, as shown in Figure 1. The trained model is able to compress the inputs into a latent representation with a smaller dimension. VAEs extend regular autoencoders by substituting the exact inference of the likelihood by the lower bound of the log-likelihood, given by the evidence lower bound (ELBO):

$$\begin{aligned} \log p_\theta(\mathbf{x}) \geq\ & \mathcal{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \\ & D_{KL}[q_\phi(z|x)||p(z)] \\ \triangleq\ & \mathcal{L}(x; \theta, \phi), \end{aligned} \tag{1}$$

where x is the observed variable, z is the latent variable with prior $p(z)$ and a conditional distribution $p_\theta(x|z)$, $q_\phi(z|x)$ is an approximation to the true posterior distribution $p_\theta(z|x)$. $q_\phi(z|x)$ and $p_\theta(x|z)$ are neural networks parametrized by ϕ and θ (encoder and decoder, respectively). D_{KL} is the Kullback–Leibler divergence.

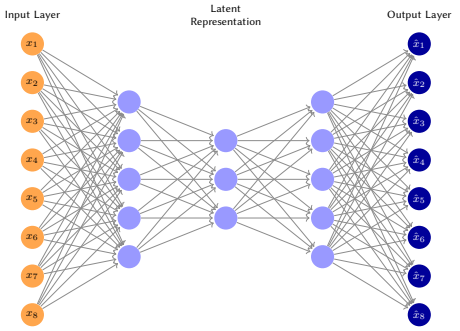


Figure 1: Example of an autoencoder network.

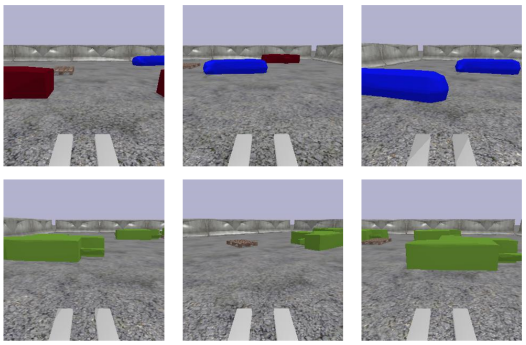


Figure 2: Examples of ID and OOD obstacles (top images and bottom images respectively). In the ID scenario, the obstacles are blue and dark red, while the OOD obstacles are green.

3.3. Uncertainty estimation based on Variational Auto Encoders

OOD detection using VAEs assumes that the model assigns higher likelihoods to the samples drawn from the in-distribution (ID) pool than the OOD samples, which is valid for different benchmarks as shown in [12]. Metrics derived from the model likelihood are then used as uncertainty estimates. We follow the Evidence Lower Bound (ELBO) Ratio method proposed in the same paper, which represents the ratio of lower bounds of the log-likelihood of a given sample and the maximum ELBO obtained with the ID samples [12]. For notation simplification, considering a fixed VAE model parametrized by ϕ and θ, the ELBO value $\mathcal{L}(x;\theta,\phi)$ will be represented as $ELBO(x)$, with $ELBO_I(x)$ representing the ELBO for a VAE model only trained with ID samples. Following this notation, the ELBO Ratio uncertainty $\mathcal{U}(x_0)$ for an arbitrary input x_0 is shown in equation 2.

$$\mathcal{U}(x_0) = \frac{ELBO(x_0)}{ELBO_I(x_{max})}, \tag{2}$$

where $ELBO_I(x_{max})$ is the maximum $ELBO$ value calculated for all ID samples (a sort of calibration based on the training data).

4. Experiments and Preliminary Results

Environment: To better support the proposed idea, experiments were conducted, and the preliminary results will be presented as further evidence. For the experiments, a custom environment was created using PyBullet [29]. It was designed to represent a warehouse with a configurable layout limited by walls, goods to be transported by an automated guided vehicle (AGV), and a set of obstacles that might be in the way. The goal is to reach a certain location that contains a good to be transported, represented by a wooden pallet, while avoiding obstacles or hitting the walls.

An RGB camera is attached to the AGV and its control decisions are made based on the state s_t encoded by the input images and the coordinates of the AGV and the goal. The image resolution can be configured, but for the results shown below, RGB images with 84 x 84 pixels were used. The observation encoding also includes the positions of the AGV and the goal. The AGV action is a 2-dimensional vector, u_t, representing the linear and angular velocities. A reward of 100 is given if the agent reaches the goal position, -100 if it hits an obstacle, and -10 if it times out (i.e., it reaches the maximum number of steps).

To attest to the capacity of the uncertainty estimator to spot critical failures that might be related to OOD instances, an ID and an OOD environment were designed. The differences consist of the type of static obstacles present in each environment, with obstacles that differ in color and shape, as shown in figure 2.

AGV controller framework: The controller used to solve the motion planning described above is shown in figure 3. The first module is a path planner, responsible to determine the optimal path to reach the goal position based on the agent's location. The planner takes the AGV kinematic model and solves the planning with the G^1 Hermite Interpolation Problem with clothoids. Interpolating a sequence of waypoints using clothoid splines will result in a smooth trajectory, suitable for the motion planning of mobile robots, as shown in [30, 31]. The planner takes a simplified observation $\tilde{s}_t$, consisting of the AGV and goal coordinates, as input. Its output is a position in the polar coordinate system $p_t = (\rho_t, \theta_t)$, where ρ_t and θ_t are the radial and angular coordinates at time t, respectively. Note that the planner does not account for obstacles, since it is assumed that obstacles are not known *a priori* and the RL agent should be re-

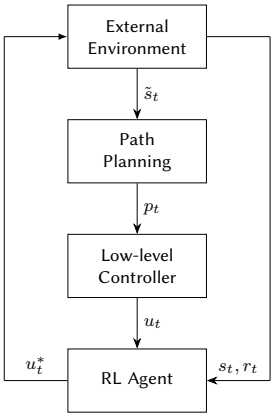
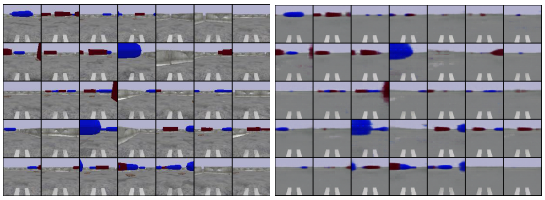
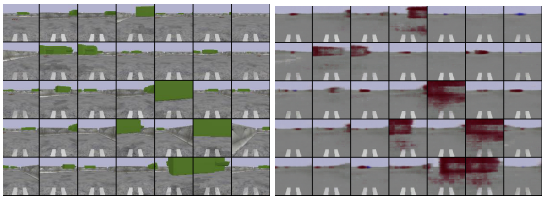


Figure 3: RL-based controller framework.



(a) ID input images. (b) ID reconstructed images.

Figure 4: VAE model compression-decompression capabilities with ID images after 10 epochs of training.



(a) OOD input images. (b) OOD reconstructed images.

Figure 5: VAE model compression-decompression capabilities with OOD images after 10 epochs of training.

sponsible to react and adjust if an unexpected obstacle is in the way. The second module is a non-linear controller used to calculate the control action u_t necessary to reach the coordinate p_t. The last module is the RL agent. Its goal is to follow the proposed trajectory, i.e., keeping $u_t \approx u_t^*$ as much as possible, proposing a different control action $u_t^* \neq u_t$ only to avoid a collision. To fulfil this task, an intrinsic reward r_{i_t} was added, with $r_{i_t} = 0.0$ if $u_t^* = u_t$ (a small difference is tolerated) and $r_{i_t} = -0.1$ otherwise. The optimal policy becomes a tradeoff between avoiding the risk of collision (with the expressive -100 reward as punishment) and following the path planner to avoid the small punishments. The RL agent was trained in the ID environment using the Soft Actor-Critic algorithm [32].

Uncertainty estimator: The VAE uncertainty estimation model was trained to fit instances randomly sampled from the ID environment in a Supervised Learning manner. To that end, 20.000 images were collected from the ID environment and 2.000 from the OOD, which are used for validation purposes during the model training. The model was trained for 10 epochs.

After training the RL agent and the VAE uncertainty estimator, rollouts are performed in the OOD environment with this agent, and (state, action, reward) tuples are saved for post-analysis. The episode termination states are then passed through the uncertainty estimator to verify if crashes present a significant correlation to high uncertainty levels. The hypothesis is that if a crash happens due to the agent not being able to avoid an obstacle semantically different from the ones experienced during training, the OOD detector could flag this instance before the crash occurs. ID inputs on the other hand should signal low uncertainty, indicating that the RL agent is able to handle such situations. It is worth mentioning that these experiments only consider a very limited number of distinguishing features for the OOD obstacles. Since in reality the number of unknown obstacles can be extremely high, these experiments should be extended to a set of obstacles that is statistically significant to the problem dimension.

Figure 4 shows how the VAE learns to reconstruct the images observed in the environment populated with ID obstacles, with the input and reconstructed images. After 10 epochs of training, the obstacles are recovered with a good definition. However, the model is not able to reconstruct the floor textures completely, which is of minor relevance in this scenario but should be investigated if such features would represent safety-critical aspects (e.g., oil in the floor, large cracks or holes).

Figure 5 on the other hand, represents the same model trained in the ID environment trying to reconstruct images with OOD obstacles in it. It is visible that, even after 10 epochs of training, the model is not able to recover the obstacle color or shape correctly, with blurred obstacles rendered in the output. That inability to correctly compress and decompress the images with OOD obstacles is responsible for increasing the calculated uncertainty.

Figure 6 shows the obtained results for the RL agent running in the OOD environment. The agent ran for 10.000 steps, which was equivalent to around 70 episodes. The y-axis represents the ELBO Ratio, which was normalized to get the values in the interval [0,1]. Episodes that ended with a crash are represented by the red bars while the blue bars picture the remaining episodes. The results show that some crash episodes presented high uncer-

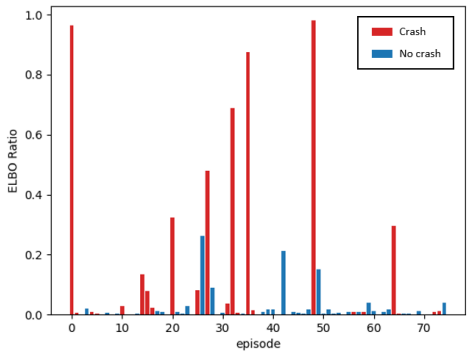


Figure 6: Uncertainty estimates on terminating states of episodes for the OOD environment.

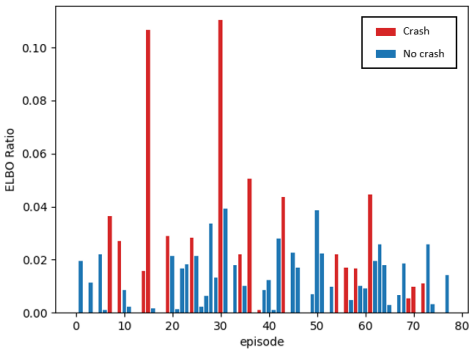


Figure 7: Uncertainty estimates on terminating states of episodes for the ID environment.

tainty, while very few non-crash episodes presented significant uncertainty levels. On the other hand, some failures did not trigger a high uncertainty level. These states could represent residual insufficiencies of the trained RL agent (e.g., caused by lack of training), that the OOD detector is not accurate for these inputs, or that the collision was not caused by an OOD element (e.g., the AGV crashed to a wall). To attest to the calibration of the uncertainty quantification, the same experiment was repeated in the ID environment, with the results shown in figure 7. The ELBO Ratio values are much lower for the entirety of the episodes and more consistent. That is expected, since in this case all the states should be considered ID, showing that the VAE is not outputting false positives for these data samples.

5. Discussion and Future Perspective

This paper focuses on motivating the promising perspective of using uncertainty quantification for improving the safety case of RL systems deployed in industrial applications, concentrating on camera-based systems. For that end, an environment modeling a typical warehouse was created. The preliminary results obtained with a VAE-based uncertainty estimator suggest this monitor can distinguish some of the states that result in accidents related to environmental distributional shifts. However, it is important to notice that not all accidents are caused by OOD obstacles, but can rather be influenced by the reward function definition, observation encoding, model generalization capabilities, among other aspects. Identifying and separating accidents caused by the inability of the agent to handle novel obstacles from accidents caused by other unrelated limitations is necessary before assessing the effectiveness of the OOD detection monitor.

Many published works already discuss the importance of uncertainty estimation and OOD detection in the whole Safe AI spectrum, but we believe a more structured way to integrate these systems and empirical results to create a compelling safety assurance case is needed, especially for RL systems. To reach this long-term goal, we suggest the following future steps:

- **Operational Design Domain (ODD) [33]:** In real-world applications, the number of contextual combination possibilities makes any attempt for extensive testing intractable. Therefore, precise system specification is paramount before starting to build the assurance case. The ODD should include all contextual information that covers the intended operation of the system.
- **Extensive experimentation:** Once an appropriate ODD is derived, the experiments described in this document can be extended to a much broader scope. Varying parameters, changing scenario configuration, considering more obstacles, and adding sensor noise are just a few aspects that should be extensively considered. Strong safety arguments will depend on the experiments achieving a high statistical confidence level for the contexts described in the ODD. This should also include multiple uncertainty estimation methods, not covered in this paper.
- **Qualitative analysis:** Understanding the system at a higher level of abstraction is also important to build a strong safety case. For that, it is important to visualize the scenarios that lead to high or low uncertainty and try to understand patterns that lead to wrong predictions, outliers, false positives and negatives, etc.
- **Residual error:** The uncertainty monitor is not intended to cover every safety aspect, but rather covers failures caused by the inability of the system to handle domain shifts. Therefore, risks

associated with other aspects will still be present and should be addressed by other methods.

- **Integration of uncertainty monitor and RL agent:** This paper focuses on how OOD scenarios might lead to system failures and how OOD detection can help in detecting such states before the failure happens. However, an important question is not addressed here and should be a high priority next step: *what to do when an OOD input is detected*? In other words, how to integrate OOD detection and a safe fallback policy into the decision-making system.
- **Failure rate calibration:** The uncertainty values are not sufficient to estimate a failure probability because an OOD instance does not necessarily imply a failure will happen. However, upper bound probabilities could be derived from the uncertainty estimates, i.e., if the model predicts that there is a 30% probability of the s_t being OOD, the risk of failures caused by distributional shifts should be below 30%.
- **SOTIF:** As shown in Section 2, traditional functional safety standards fail to properly address ML systems. In contrast, SOTIF is a much more appropriate framework to build a safety argumentation for such cases. However, building an assurance case based on an uncertainty-aware RL agent, to the best of our knowledge, was not yet done. In SOTIF it is necessary to attest to the absence of unreasonable risk due to hazards resulting from functional insufficiencies of the intended functionality, which is challenging due to the nature of model-free RL and sequential decision-making systems in general.

Not necessarily those items were touched on in this paper, but this list serves as a roadmap to guide our research efforts in the near future, as we believe that covering these points in deeper detail will result in incremental progress towards achieving a sound argumentation to enable uncertainty-aware RL agents to be deployed in safety-critical applications.

Acknowledgments

This work was funded by the Bavarian Ministry for Economic Affairs, Regional Development and Energy as part of a project to support the thematic development of the Institute for Cognitive Systems.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602 (2013).

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, nature 550 (2017) 354–359.

[3] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al., Dota 2 with large scale deep reinforcement learning, arXiv preprint arXiv:1912.06680 (2019).

[4] C. Esposito, X. Su, S. A. Aljawarneh, C. Choi, Securing collaborative deep learning in industrial applications within adversarial scenarios, IEEE Transactions on Industrial Informatics 14 (2018) 4972–4981.

[5] R. A. Khalil, N. Saeed, M. Masood, Y. M. Fard, M.-S. Alouini, T. Y. Al-Naffouri, Deep learning in the industrial internet of things: Potentials, challenges, and emerging applications, IEEE Internet of Things Journal 8 (2021) 11016–11040.

[6] M. Maqsood, I. Mehmood, R. Kharel, K. Muhammad, J. Lee, W. Alnumay, Exploring the role of deep learning in industrial applications: a case study on coastal crane casting recognition, Hum. Cent. Comput. Inf. Sci 11 (2021) 1–14.

[7] J. M. P. Schumann, Y. Liu, Applications of neural networks in high assurance systems, volume 268, Springer, 2010.

[8] F. Schwaiger, M. Henne, F. Küppers, F. S. Roza, K. Roscher, A. Haselhoff, From black-box to white-box: Examining confidence calibration under different conditions, arXiv preprint arXiv:2101.02971 (2021).

[9] A. Filos, P. Tigkas, R. McAllister, N. Rhinehart, S. Levine, Y. Gal, Can autonomous vehicles identify, recover from, and adapt to distribution shifts?, in: International Conference on Machine Learning, PMLR, 2020, pp. 3145–3153.

[10] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, M. Hardt, Test-time training with self-supervision for generalization under distribution shifts, in: International conference on machine learning, PMLR, 2020, pp. 9229–9248.

[11] S. Thulasidasan, S. Thapa, S. Dhaubhadel, G. Chennupati, T. Bhattacharya, J. Bilmes, An effective baseline for robustness to distributional shift, in: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2021, pp. 278–285.

[12] X. Ran, M. Xu, L. Mei, Q. Xu, Q. Liu, Detecting out-of-distribution samples via variational auto-encoder with reliable uncertainty estimation, Neural Networks (2022).

[13] R. Ashmore, R. Calinescu, C. Paterson, Assuring the

Machine Learning Lifecycle: Desiderata, Methods, and Challenges (2019).
[14] R. Salay, R. Queiroz, K. Czarnecki, An Analysis of ISO 26262: Using Machine Learning Safely in Automotive Software (2017).
[15] O. Willers, S. Sudholt, S. Raafatnia, S. Abrecht, Safety Concerns and Mitigation Approaches Regarding the Use of Deep Learning in Safety-Critical Perception Tasks (2020).
[16] S. Burton, J. A. McDermid, P. Garnett, R. Weaver, Safety, Complexity, and Automated Driving: Holistic Perspectives on Safety Assurance, Computer 54 (2021).
[17] A. Kendall, Y. Gal, What uncertainties do we need in bayesian deep learning for computer vision?, Advances in neural information processing systems 30 (2017).
[18] E. Hüllermeier, W. Waegeman, Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods, Machine Learning (2021).
[19] M. Henne, A. Schwaiger, G. Weiss, Managing uncertainty of ai-based perception for autonomous systems., in: AISafety@ IJCAI, 2019, pp. 11–12.
[20] A. Schwaiger, P. Sinhamahapatra, J. Gansloser, K. Roscher, Is uncertainty quantification in deep learning sufficient for out-of-distribution detection?, in: AISafety@ IJCAI, 2020.
[21] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: international conference on machine learning, PMLR, 2016, pp. 1050–1059.
[22] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, Advances in neural information processing systems 30 (2017).
[23] M. Sensoy, L. Kaplan, M. Kandemir, Evidential deep learning to quantify classification uncertainty, Advances in neural information processing systems 31 (2018).
[24] T. Yang, H. Tang, C. Bai, J. Liu, J. Hao, Z. Meng, P. Liu, Exploration in deep reinforcement learning: a comprehensive survey, arXiv preprint arXiv:2109.06668 (2021).
[25] T. Haider, F. S. Roza, D. Eilers, K. Roscher, S. Günnemann, Domain shifts in reinforcement learning: Identifying disturbances in environments., in: AISafety@ IJCAI, 2021.
[26] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, S. Levine, Uncertainty-aware reinforcement learning for collision avoidance, arXiv preprint arXiv:1702.01182 (2017).
[27] B. Lütjens, M. Everett, J. P. How, Safe reinforcement learning with model uncertainty estimates, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019, pp. 8662–8668.
[28] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).
[29] E. Coumans, Y. Bai, Pybullet, a python module for physics simulation for games, robotics and machine learning (2016).
[30] E. Bertolazzi, M. Frego, G1 fitting with clothoids, Mathematical Methods in the Applied Sciences 38 (2015) 881–897.
[31] P. Bevilacqua, M. Frego, E. Bertolazzi, D. Fontanelli, L. Palopoli, F. Biral, Path planning maximising human comfort for assistive robots, in: 2016 IEEE Conference on Control Applications (CCA), IEEE, 2016, pp. 1421–1427.
[32] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, in: International conference on machine learning, PMLR, 2018, pp. 1861–1870.
[33] K. Czarnecki, Operational design domain for automated driving systems, Waterloo Intelligent Systems Engineering (WISE) (2018).