Co-design of out-of-distribution detectors for autonomous emergency braking systems

Yuhas, Michael; Easwaran, Arvind

2023

Yuhas, M. & Easwaran, A. (2023). Co-design of out-of-distribution detectors for autonomous emergency braking systems. 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), 1996-2003.
https://dx.doi.org/10.1109/ITSC57777.2023.10421953

https://hdl.handle.net/10356/169620

https://doi.org/10.1109/ITSC57777.2023.10421953

Co-Design of Out-of-Distribution Detectors for Autonomous Emergency Braking Systems*

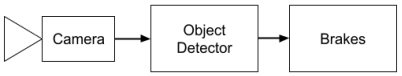
Michael Yuhas[1,2] and Arvind Easwaran[2]

***Abstract*— Learning enabled components (LECs), while critical for decision making in autonomous vehicles (AVs), are likely to make incorrect decisions when presented with samples outside of their training distributions. Out-of-distribution (OOD) detectors have been proposed to detect such samples, thereby acting as a safety monitor, however, both OOD detectors and LECs require heavy utilization of embedded hardware typically found in AVs. For both components, there is a tradeoff between non-functional and functional performance, and both impact a vehicle's safety. For instance, giving an OOD detector a longer response time can increase its accuracy at the expense of the LEC. We consider an LEC with binary output like an autonomous emergency braking system (AEBS) and use risk, the combination of severity and occurrence of a failure, to model the effect of both components' design parameters on each other's functional and non-functional performance, as well as their impact on system safety. We formulate a co-design methodology that uses this risk model to find the design parameters for an OOD detector and LEC that decrease risk below that of the baseline system and demonstrate it on a vision based AEBS. Using our methodology, we achieve a 42.3% risk reduction while maintaining equivalent resource utilization.**

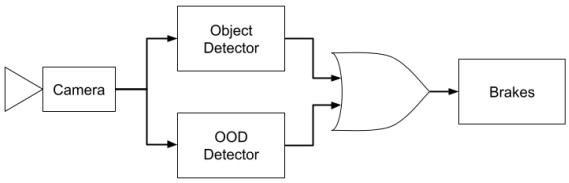
I. INTRODUCTION

When learning enabled components (LECs) are exposed to data outside their training distribution, they are unlikely to make correct decisions. In safety critical systems like autonomous vehicles, such a failure could lead to catastrophic results. Out-of-distribution detectors have been proposed to detect such data [1], however, the introduction of an OOD detector exposes the system to additional risks. If the OOD detector does not yield a decision before its deadline or returns a false negative result, it provides no protection. Furthermore, when the OOD detector shares the same computational resource (like an embedded GPU) with an LEC, it interferes with the LEC's ability to meet deadlines [2]. Additionally, false positives from the OOD detector can affect the system's availability, leading to a decrease in performance [3].

Given these challenges, we seek to co-design an OOD detector and an LEC such that the new system (Fig. 1b) uses the same hardware platform as the original design (Fig. 1a), but positively impacts safety. As both the OOD detector and the LEC are implemented with deep neural networks (DNNs), we identify a subset of hyperparameters that can be selected independently for both components that tradeoff functional and non-functional performance. However, this problem is different from a typical hyperparameter selection problem. First, the objective is a function of both deadline misses and the functional performance of each network; it is not enough to minimize one of these values constrained on the others. For example, a DNN with higher accuracy may be able to tolerate a greater level of deadline misses. Second, the parameter selection for the each of the networks is not independent. For example, choosing a parameter for the OOD detector that lengthens execution time and increases accuracy will affect the response time of the LEC.



(a) Baseline AEBS; only the object detector triggers braking.



(b) OOD detector acts as a safety monitor for the object detector.

Fig. 1: Block diagram of the AEBSs considered in this paper.

We use risk as an objective to combine the functional and non-functional performance of both components and express the system risk for a generic binary classification task in terms of its components' failure probabilities. We then apply this model to a simplified autonomous emergency braking system (AEBS), which functions as a binary classifier (emergency stop or no action) for a given input sample. The contributions of this paper are as follows:

- We derive a risk model valid for any binary classification task that expresses risk in terms of the co-design hyperparameters of an LEC and an OOD detector.
- We formulate a co-design methodology that efficiently explores the design space of OOD detector and LEC hyperparemeters to minimize risk without exceeding the average utilization of the baseline system.
- We co-design an OOD detector and object detector for a vision based AEBS to demonstrate OOD detectors' potential for risk reduction when deployed intelligently. We show up to a 42.3% risk reduction while maintaining the same average resource utilization as the baseline.

*This research was funded in part by MoE, Singapore, Tier-2 grant number MOE2019-T2-2-040. This research is part of the programme DesCartes and is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

[1]Energy Research Institute @ NTU, Interdisciplinary Graduate Programme, Nanyang Technological University, Singapore

[2]School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798 {michaelj004,arvinde}@e.ntu.edu.sg

II. BACKGROUND

A. Out-of-Distribution Detection

Building functional OOD detectors has been well studied. Detectors like ODIN [4] use the intermediate layers of a DNN to identify OOD samples. The disadvantage is that the functional performance of the OOD detector is tightly coupled with the LEC it is monitoring. In [5], An and Cho proposed an OOD detector whose weights are trained independently and therefore conditionally independent of an LEC given a shared training set. They used the reconstruction loss of a variational autoencoder (VAE) to determine whether or not a given sample was drawn from the training distribution. In [1], Cai and Koutsoukos demonstrated the effectiveness of the reconstruction based OOD detector for autonomous driving and used inductive conformal prediction (ICP) and a martingale to deal with the time dimension. However, reconstructing an image via VAE is costly in terms of execution time. In [6], OOD detection in the latent space of a VAE was proposed, which only required running the encoder portion of the VAE. Ramakrishna *et al.* showed that the latent space of a VAE could be partially disentangled to detect OOD samples from different generative factors (e.g., rain, brightness, etc.), and demonstrated this on an autonomous driving dataset [7]. Although design methodologies have been proposed to optimize the execution time of such an OOD detector while respecting bounds on accuracy [8], they do not take into account the scenario where an OOD detector and an LEC share the same set of computational resources.

B. Co-Design frameworks for LECs

Previous literature has focused on risk analysis and risk reduction as a means to safely deploy LECs in AVs. In [9], the ReSoNate framework was proposed, which calculated run time risk given environmental state and internal state and used this information to select the best controller for a particular scenario. Although this framework used OOD detectors for safety monitoring, it did not account for the effect of an OOD detector on the LEC's response time. In [10]–[12], traditional risk analysis techniques were applied to automotive systems containing LECs, but no concrete methodology to apply them toward the co-design problem was demonstrated. In [3], Alecu *et al.* described the problem of LEC and monitor design as a tradeoff between safety and availability, however, they only explored this tradeoff in terms of functional performance, not response times.

Other works have focused on the deployment of multiple DNNs to shared resources while respecting schedulability. In [13], an algorithm was presented for scheduling multiple DNNs across a CPU and GPU that took into account accuracy loss due to quantization. While the deadlines of both tasks were respected, the algorithm could not actively trade accuracy between its scheduled tasks to minimize a shared risk objective. In [14], DNNs were also divided across resources and their accuracy bounds were checked before scheduling. A neural architecture search could improve the accuracy of a DNN if response time allowed, but it could not take into account the cumulative effect of multiple DNNs.

III. PROBLEM DEFINITION

A. Risk Minimization over Design Parameters

We consider a system with two DNNs like the AEBS in Fig. 1b. We refer to the object detector as the *essential component* (EC), and provide a generalized risk model for any EC that implements a binary classification task. Let the EC be defined as $f(x;\theta_{EC},\lambda_{EC},\tau_{EC}) : \mathbb{R}^n \mapsto \{0,1\}$. Here, x is an n dimensional input sample, e.g., radar point cloud or image. The EC maps this to one of two values: a 0 indicates a negative result and a 1 indicates a positive result. f is parameterized by λ_{EC}, a set of hyperparemeters determining the structure of f, e.g., number of layers or input image size; θ_{EC}, a set of parameters learned during training; and τ_{EC}, the confidence threshold at which the EC returns a 1. Likewise, let the OOD detector be defined as $g(x;\theta_{OOD},\lambda_{OOD},\tau_{OOD}) : \mathbb{R}^n \mapsto \{0,1\}$. Once again x is an n dimensional input data sample that the OOD detector maps to 0 for in-distribution (ID) or 1 for OOD. The OOD detector is parameterized by λ_{OOD}, a set of hyperparameters determining the structure of g; θ_{OOD}, the parameters learned during training; and τ_{OOD}, the confidence threshold at which the OOD detector returns a 1. Let $\Lambda = (\lambda_{EC},\lambda_{OOD})$ be a tuple of the hyperparameters that affect the response times and functional performance of both components, while $T = (\tau_{EC},\tau_{OOD})$ and $\Theta = (\theta_{EC},\theta_{OOD})$ affect functional performance only.

It is impossible to design an OOD detector or EC with zero failure rate. We are interested in the negative effects that occur if these components fail, how likely failures are to occur, and how severe their consequences would be. Risk, a combination of the severity of an event with its probability is a natural way to measure this, and is found in automotive safety standards like ISO 26262 [15]. We define risk mathematically in (1), where $\mathcal{R}$ is the system's total risk, $\mathcal{E}$ is the set of all hazardous events that can occur, $P(x)$ denotes the probability of event x, and $S(x)$ denotes its severity.

$$\mathcal{R} = \sum_{E_i \in \mathcal{E}} P(E_i)S(E_i) \tag{1}$$

Our goal is to design a system that minimizes risk as defined in (1), such that the average resource utilization does not exceed that of the baseline as shown in (2).

$$\begin{aligned} &\underset{\Lambda,T,\Theta}{\operatorname{argmin}} \sum_{E_i \in \mathcal{E}} P(E_i|\Lambda,T,\Theta)S(E_i) \\ &\text{s.t.} \quad \bar{U}(\Lambda) \leq \bar{U}_{base} \end{aligned} \tag{2}$$

Here, $\bar{U}(\Lambda)$ is the average resource utilization of a system given structural parameters Λ and $\bar{U}_{base}$ is the average resource utilization of the baseline system (e.g., Fig. 1a). We define $\bar{U}$, average resource utilization, as the percentage of time the shared computational resource is occupied in one period (from both jobs' release times to their deadline) averaged over all periods. Constraining utilization is reasonable because the EC plus OOD detector should be a drop-in replacement for the baseline system; if the baseline utilization is exceeded, it may interfere with other system-level

TABLE I: ASSUMPTIONS UNDER WHICH OUR RISK MODEL HOLDS.

Asm.	Description
A1	Negligible risk for correct classification
A2	No reject option
A3	Hazard avoidance when EC *or* OOD detector returns 1
A4	Results of both components are independent across time (*i.e.,* dependency across successive inputs is ignored)
A5	If EC or OOD detector does not complete before its deadline, all work is discarded and execution is terminated
A6	No action is triggered due to early termination
A7	OOD detector missing a deadline is independent of its detection result
A8	Functional results of the EC and OOD detector are conditionally independent given $\neg E_e$ or $\neg E_\epsilon$

components. Note that by changing the design parameters Λ, T, and Θ, we can affect the probability of an event occurring, but we cannot change its severity. While the learned parameters Θ affect the probability of an event occurring, minimizing risk subject to these learned parameters is outside the scope of this work as it can be optimized through existing training methods like [16]. In order to solve this minimization problem, we need to identify the events that compose $\mathcal{E}$. Our assumptions are listed in Table I. Under A1, the risk induced by the EC making a correct decision (true positive or true negative) is negligible and has already been minimized under safety of the intended functionality [17]. Under A2, the system must generate one of two outputs: 0 or 1; there is no reject option. This assumption is valid as even if the OOD detector correctly identifies an OOD input, the system must perform some action, even if that action is to keep doing the same thing while waiting to reprocess the rejected sample with another model [3]. Under A3, we assume that either a binary classification result of 1 *or* an OOD detection result of 1 will trigger a hazard avoidance action. This reflects the use of OOD detectors in prior works [1], [2], [9] and leads to two possible failure modes, $\mathcal{E} = \{E_0, E_1\}$, where:

- E_0 – The event where no action is taken when a hazard is present (OOD detector and EC return 0 while the ground truth for the EC is 1)
- E_1 – The event where corrective action is taken when no hazard is present (OOD detector returns 1 *or* EC returns 1 while the ground truth for the EC is 0)

B. Determining Probabilities through Fault Tree Analysis

We use fault tree analysis (FTA), a deductive method that works backwards from a given failure mode and determines which intermediate failures must occur to cause it. FTA allows us to express the probabilities of the top-level events in terms of the probabilities of their generating events [18]. We want to express the probabilities of E_0 and E_1 with respect to the design parameters Λ, T, and Θ. Under A4, we assume the independence of detection results on different samples over time. Although not strictly true in practice as observations in one time instance depend on the control action from the previous instance, we are dealing with feed forward DNNs, so previous results are not used directly in the computation of the next result. Other works have made this assumption as well [19].

TABLE II: SAMPLE SPACE FOR THE OUTCOME OF THE EC.

Event	Definition
E_a	The event that the EC gives a false positive result
E_b	The event that the EC gives a true positive result
E_c	The event that the EC gives a false negative result
E_d	The event that the EC gives true negative result
E_e	The event that the EC misses its deadline

TABLE III: SAMPLE SPACE FOR THE OUTCOME OF THE OOD DETECTOR.

Event	Definition
E_α	The event that the OOD detector gives a false positive result
E_β	The event that the OOD detector gives a true positive result
E_γ	The event that the OOD detector gives a false negative result
E_δ	The event that the OOD detector gives true negative result
E_ϵ	The event that the OOD detector misses its deadline

First, we define the intermediate events that could lead to the occurrence of E_0 and E_1. Each execution of the EC can be considered as an experiment where the outcome is one of the events in Table II. Note that E_a, E_b, E_c, E_d, and E_e are mutually exclusive and $P(E_a) + P(E_b) + P(E_c) + P(E_d) + P(E_e) = 1$. Likewise, Table III shows the events representing the possible outcomes of the OOD detector. Similar to the previous case, E_α, E_β, E_γ, E_δ, and E_ϵ are mutually exclusive and $P(E_\alpha) + P(E_\beta) + P(E_\gamma) + P(E_\delta) + P(E_\epsilon) = 1$. Under A5, we assume that as soon as any component misses its deadline for a sample x, all work is discarded and the execution is terminated; this assumption is also considered in other literature [20]. Under this assumption, while the OOD detector and EC missing a deadline for the same input are dependent events ($E_e \not\perp E_\epsilon$), these events are independent across samples. Under A6, we assume that missing a deadline will not cause any corrective action. This is a common strategy in controls literature [21] and we use it here to reduce false positives. Let E_0^{base} and E_1^{base} correspond to E_0 and E_1 in the baseline system. The fault tree analysis trivially yields (3) and (4).

$$E_0^{base} = E_c \vee E_e \implies P(E_0^{base}) = P(E_c) + P(E_e) \quad (3)$$

$$E_1^{base} = E_a \implies P(E_1^{base}) = P(E_a) \quad (4)$$

For the system with OOD detector, let E_0^{mod} and E_1^{mod} correspond to E_0 and E_1:

$$E_0^{mod} = \{E_\gamma \vee E_\delta \vee E_\epsilon\} \wedge \{E_c \vee E_e \wedge E_{pos}\} \quad (5)$$

$$E_1^{mod} = \{\{E_\beta \vee E_\alpha\} \wedge \neg E_{pos}\} \vee E_a \quad (6)$$

Here, E_{pos} is the event that a sample's ground truth is 1. Note that the OOD detector reduces the chance of E_0^{mod} as a functional failure or deadline miss of both components is required. However, the OOD detector increases the chance of E_1^{mod}, as any positive result leads to this failure as long as the ground truth is 0. Under A7, we assume that the OOD detector's probability of missing a deadline is not a function of the sample's ground truth. This assumption is based on the architecture of the OOD detectors we consider [7]. We do not make the same assumption for the EC, as some architectures may take different amounts of time to reach a decision depending on the input data. For example, in our AEBS use case, a YOLO object detector requires non-max suppression to select the best bounding boxes, so its execution time is

dependent on the number of boxes identified [22]. Although the results of the OOD detector and the EC are dependent due to response time interference, under A8, we assume that given no deadline miss occurs, the functional results of both components are independent. We make this assumption due to the use of an independently trained OOD detector like [5].

$$P(E_x|E_y) = P(E_x|\neg E_e)\forall x \in \{\alpha,\beta,\gamma,\delta\}; y \in \{a,b,c,d\}$$
$$P(E_x|E_y) = P(E_x|\neg E_\epsilon)\forall x \in \{a,b,c,d\}; y \in \{\alpha,\beta,\gamma,\delta\} \quad (7)$$

The equalities in (7) arise from A8 and we use them to simplify $P(E_0^{mod})$ (8) and $P(E_1^{mod})$ (9).

$$\begin{aligned} P(E_0^{mod}) = P(Ec)(&P(E_\gamma) + P(E_\delta) + P(E_\epsilon) \\ &- P(E_\gamma|E_e) - P(E_\delta|E_e) - P(E_\epsilon|E_e)) \\ &+ P(E_{pos})P(E_e|E_{pos})(P(E_\gamma|E_e) \\ &+ P(E_\delta|E_e) + P(E_\epsilon|E_e)) \end{aligned} \quad (8)$$

$P(E_0^{mod})$ is composed of two terms: the first takes into account the false negative rate of the EC and the probability that the OOD detector fails to override it while the second considers the case when the EC misses its deadline and the OOD detector fails to override. Notice that an estimate of $P(E_{pos})$ is required to calculate this probability.

$$\begin{aligned} P(E_1^{mod}) = P(E_a) &+ P(E_\alpha)(1 - P(E_{pos})) \\ &+ P(E_\beta)(1 - P(E_{pos})) - P(E_a)(P(E_\alpha) \\ &+ P(E_\beta) - P(E_\alpha|E_e) - P(E_\beta|E_e)) \end{aligned} \quad (9)$$

$P(E_1^{mod})$ is composed of three terms. Either a false positive from the EC, or a positive result from the OOD detector when the ground truth is 0 (terms one and two) can trigger this event. The subtraction of term three removes the union of these two events as they are not mutually exclusive.

IV. CO-DESIGN METHODOLOGY

The constituent terms of (8) and (9) can be estimated empirically for an EC and OOD detector pair with unique set of parameters Λ, T, and Θ. However, evaluating the risk for one pair involves training two separate DNNs, which means exploring the design space of all parameter combinations is prohibitively expensive, so we propose a design methodology to minimize the training time needed to find a satisfactory solution. We note the following facts about our minimization problem, which inform our design strategy. Firstly, for fixed hyperparameters Λ and T, the resulting probability of an event $P(E_i)$ is not fixed. This is because training a DNN is a stochastic process, and retraining will result in a different set of learned parameters Θ. Next, we note that Λ can be comprised of both numerical and categorical variables. An example of a numerical variable is the scaled input size to the EC or the OOD detector, while an example of a categorical variable is the architecture used for a particular component (e.g., a β-VAE OOD detector or a reconstruction based OOD detector). The presence of categorical variables means that gradient based optimization techniques are not an option. While it is desirable to evaluate as few values of Λ and Θ as necessary, once we have trained two DNNs for a (Λ, Θ) pair, it is relatively easy to find the T that minimizes risk for that configuration. This is because during testing, the EC and OOD detector will produce a list of confidence scores for a given dataset and the functional performance at multiple thresholds can be computed without the need for the DNNs to re-infer the entire training set.

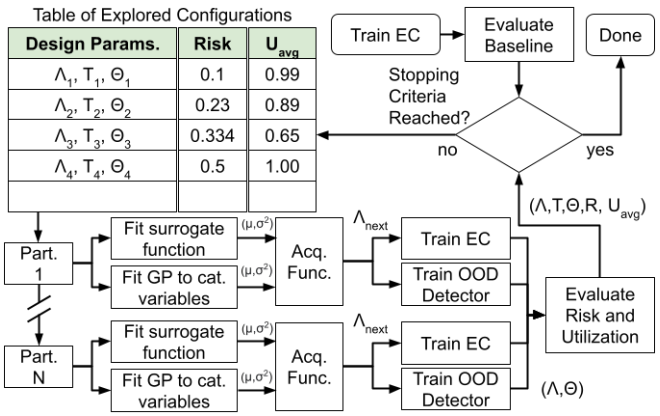


Fig. 2: Risk-aware co-design methodology for OOD detector and EC based on Bayesian optimization.

Since we are dealing with a possibly non-convex, noisy risk function, we propose a modified version of Bayesian optimization [23] to find the parameters Λ and T, that minimize risk. A visualization of our design methodology is shown in Fig. 2. A table keeps track of all the design parameter combinations that have been tried and their respective risk and average utilization values. First, the EC for the baseline system with no OOD detector is trained and the risk and average utilization are evaluated; the average utilization will serve as a constraint when evaluating proposed solutions. Next, the search space is divided into N partitions of equal size. By dividing the search space into partitions, we can help reduce the likelihood that the Bayesian optimization gets stuck at a local minimum [24]. To start, we populate the table with n_{init} entries with Λs uniformly sampled across each partition. Given the solutions already evaluated in each partition, we fit a *surrogate function*. The surrogate function estimates of the true risk value at any point as well as the confidence in that estimate. We accomplish this with one Gaussian process model for all numerical parameters in Λ [25] and use a separate Gaussian process model for each categorical variable if present [26]. Next, in each partition an *acquisition function* uses the mean and variance from each surrogate function to generate a numerical estimate for how beneficial it would be to evaluate a new sample at a given point. The choice of acquisition function affects the balance between exploiting existing good solutions and exploring other areas in the risk function's domain; we choose *expected improvement* [25]. The Λ that maximizes the acquisition function is determined numerically using the conjugate gradient method and then used to train a new EC and OOD detector. Risk and utilization are experimentally evaluated for the (Λ, Θ) pair in each partition and T is swept across the its entire range for the EC and OOD detector.

The solution at the T that minimizes risk is compared with the baseline risk and average utilization. If the utilization constraint is satisfied, the tuple $(\Lambda, T, \Theta, \mathcal{R}, \bar{U})$ is recorded in the table of previously evaluated points. However, if the constraint is violated, its risk is set to an arbitrarily high value and then recorded in the table to encourage the acquisition function to look elsewhere. We choose lack of improvement in risk after a set number of iterations as the stopping criteria. While Bayesian optimization is not guaranteed to converge or find the minimum solution, it provides a powerful tool to deal with such a design problem.

V. CASE STUDY: YOLO BASED AEBS

In order to show the effectiveness of the proposed risk minimization strategy, we conduct a case study on a YOLO (You Only Look Once) based AEBS with OOD detector. This AEBS uses monocular vision to identify obstacles as demonstrated in [27]. The output of the YOLO object detector is processed to make a binary decision: 0 (do nothing) or 1 (engage emergency braking). We use images with heavy rain (not present during YOLO or OOD detector training) as OOD samples. The code to generate our dataset, train the LECs, and execute the tests is publicly available[1].

A. Dataset

We use CARLA simulator version 0.9.13 [28] to simulate an autonomous vehicle in an urban environment. 100 video clips of 500 frames at 30 FPS are captured in CARLA built-in town 10 (urban environment). The ego vehicle uses the default autopilot to navigate the streets and additional vehicles and pedestrians are spawned into the town to add obstacles on the road. The ego vehicle is selected as the Audi E-Tron with a forward facing RGB camera and segmentation camera affixed at relative coordinates $(x = 10.5, y = 0, z = 0.7)$. The outputs of both cameras are resized to 800×600 pixels. The default time of day and weather are used for all the gathered clips. We use the same method as [8] to augment the images with varying amounts of rain. Image augmentation is applied such that 10 rain levels ranging from $0, 0.1, 0.2, \ldots 0.9$ are applied to 10 different clips each containing 500 images, resulting in 5000 total images at each rain level in the dataset (5×10^4 images in total). Some examples of the generated images are shown in Fig. 3.

B. Object Detector

As proposed in [27], we use YOLO to perform object detection. Specifically, we select the YOLOv7 tiny model [22], as this provides state of the art object detection performance while keeping execution time low with respect to other YOLO variants. Our free parameter λ_{EC} for the co-design is the input image size, and the confidence threshold τ_{EC} is used by YOLO to determine if a bounding box is detected as an object or not. We select 300 images from clips $0-19$ (no rain and 10% rain) as the training set (6000 images in total) and 100 images from clips $0 - 19$ as the cross-validation set (2000 images in total). The ground truth bounding boxes were automatically obtained from the segmentation camera images by finding the contours of each region and the minimum bounding box that contained them. Only the pedestrian and vehicle object classes were considered. For this study images with 10% rain or less are considered ID and images with more than 10% rain are considered OOD. The 10% cutoff was determined empirically as the point at which YOLO's performance begins to degrade.

To obtain the performance of the object detector in terms of $P(E_a)$, $P(E_b)$, $P(E_c)$, $P(E_d)$, and $P(E_e)$ we define two metrics unique to a vision based object detector. While the traditional false positive rate (FPR) of an object detector gives the rate of false positive bounding boxes per input image [29], we are only interested in the case where at least one false positive is found, because this is enough to trigger emergency braking. Furthermore, even if a false positive detection occurs in a sample with another object present, it does not lead to the event E_a, since the vehicle is supposed to stop anyway. Our modified FPR (FP_m) metric is defined in (10), where $\mathcal{D}$ is a dataset consisting of tuples (x, y); x is an input image and y is a list of objects present in the input. $f(x)$ is the trained object detector.

$$FP_m(f, \mathcal{D}) = \frac{|\{(x,y)|f(x) = 1 \wedge |y| = 0; \forall (x,y) \in \mathcal{D}\}|}{|\mathcal{D}|} \tag{10}$$

Likewise, we redefine false negative rate (FNR) as the number of inputs that contain an object, but where no bounding box is identified. Even if the object detector identifies the wrong object, this event will not contribute to E_1. Our modified FNR (FN_m) metric is defined in (11).

$$FN_m(f, \mathcal{D}) = \frac{|\{(x,y)|f(x) = 0 \wedge |y| > 0; \forall (x,y) \in \mathcal{D}\}|}{|\mathcal{D}|} \tag{11}$$

C. OOD Detector

For the OOD detector, we consider two free parameters that make up λ_{OOD}: the size of the input image and the OOD detector type (β-VAE or reconstruction based). The β-VAE OOD detector [7] is designed to identify OOD samples caused by a specific generative factor. We select 200 images from clips $0 - 19$ (no rain and 10% rain) as the proper training set (4000 images in total), and 100 images from clips $0 - 19$ for the calibration set (2000 images in total). This splitting strategy leaves us with the 2:1 train:calibration split recommended in [7]. The VAE portion of the OOD detector was constructed with four convolutional layers with depths 32/64/128/256 and a convolutional kernel of size 3, each followed by a maxpool with kernel size 2. The fully connected layers were sized 2048, 1000, 250, and finally 50 latent variables. All layers used a leaky ReLU activation function with $\alpha = 0.1$ except for the final layer which used an identity activation function. The decoder was constructed as a mirror image of the encoder. All variants of the network were trained for 350 epochs using the Adam optimizer [16] with maximum learning rate set to 10^{-5}. After training, the calibration set was used to select the latent variable that

[1]https://github.com/CPS-research-group/CPS-NTU-Public/tree/ITSC2023

Rain: 0% Rain: 20% Rain: 40% Rain: 60% Rain: 80%

Fig. 3: Example images for selected rain levels in our dataset.

responded best to rain for each model, and the Kullback-Leibler divergence for each sample in the calibration set was used as a non-conformity score to train the detector's ICP. For the reconstruction based OOD detector from [1], we used the same train:calibration split, but the OOD score is now the mean squared reconstruction error at the output of the decoder. After training, the non-conformity scores between each sample in the calibration set and all the samples in the proper training set were calculated using the k-nearest neighbors algorithm with $k = |\mathcal{D}_{train}|$, the size of the training set. These non-conformity scores were then used for ICP at the detector's output.

D. Execution Dispatch and Timing

To determine $P(E_e)$ and $P(E_\epsilon)$, we test the entire system on an Nvidia Jetson Nano with 2 GB RAM and L4T 32.1 with the PREEMPT_RT kernel patch installed. Both the YOLO object detector and OOD detector were executed on the embedded GPU, however, because the Jetson platform does not support NVIDIA MPS [30], both the detectors have to be submitted to the GPU sequentially, with the resource blocked until one is finished. The distributions $P(E_e)$ and $P(E_\epsilon)$ are generated empirically by analyzing the response times on 1000 images. To calculate average utilization we measure the execution times of each job in a period and take the average percentage of time spent working on the two jobs across all periods.

E. Optimal Baseline AEBS

First, we find the optimal design parameters for a baseline AEBS with only an object detector. We evaluate the design with deadlines every 250 ms, which corresponds to processing a video stream at 4 Hz. In Fig. 4 we plot $P(E_0^{base})$ and $P(E_1^{base})$ across the entire design space. As we expected, when the input image size is small, $P(E_0^{base})$ tends to be large as it is more difficult for YOLO to identify objects. Likewise when image sizes are large, $P(E_0^{base})$ is also high due to deadline misses. $P(E_1^{base})$ appears steady across the range of sizes indicating robustness to false positives. While $P(E_0^{base})$ is minimized with a lower threshold, $P(E_1^{base})$ is minimized with the highest threshold. By combining both events with risk, we are able to find a solution that compromises between the two metrics. Table IV (YOLO Only) shows the minimum risk and corresponding average utilization for the baseline system along with the optimal Λ and T. We calculate risk assuming $S(E_0^{base}) = 3$ as this can lead to a head on collision and $S(E_1^{base}) = 1$ as this event can cause the vehicle to be struck from the rear [31].

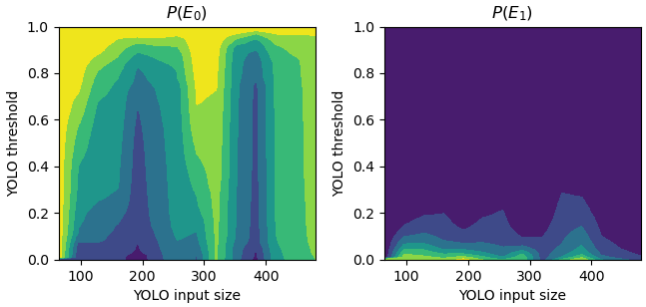


Fig. 4: $P(E_0^{base})$, $P(E_1^{base})$ across the entire design space of the baseline AEBS. Here, the deadline is set at 250 ms (4 Hz). Lighter colors indicate values approaching 1 while darker colors indicate values approaching 0.

TABLE IV: COMPARISON OF OPTIMAL DESIGN ATTRIBUTES FOR THE BASELINE AEBS AND COMBINED YOLO-OOD AEBS AT VARIOUS SAMPLING FREQUENCIES.

Config.	YOLO Only @4Hz	Combined @3Hz	Combined @4Hz	Combined @5Hz
Min. Risk	0.6337	0.2735	0.2104	Infeas.
YOLO Thresh.	0.20	0.61	0.14	Infeas.
YOLO Size.	192×192	384×384	160×160	Infeas.
OOD Thresh.	N/A	0.99	0.99	Infeas.
OOD Size.	N/A	16×16	64×64	Infeas.
OOD Arch.	N/A	β-VAE	β-VAE	Infeas.
U	0.9252	0.6646	0.9232	Infeas.

F. Risk Minimization for AEBS with OOD Detector

Given the minimum risk baseline in Table IV, we use our design strategy to select an EC / OOD detector combination that lowers risk below the baseline case while not exceeding the baseline's $\bar{U}$. We set the number of partitions to 4 as described in Table V, and set n_{init} to 5. When gathering timing data, we allow the OOD detector to always run first when a new input arrives. This design choice ensures that the OOD detector gets a chance to execute as YOLO blocks the embedded GPU for nearly the entire available duration, even at lower input sizes. Fig. 5 shows the convergence of our Bayesian optimization strategy (orange) compared with grid search (green). The grid search was conducted by

TABLE V: PARTITIONS USED IN OUR CO-DESIGN.

Part.	**YOLO Sizes**	**OOD Sizes**	**OOD Arch.**
1	64–272	8–116	{β-VAE, reconstruction based}
2	272–480	8–116	{β-VAE, reconstruction based}
3	64–272	116–224	{β-VAE, reconstruction based}
4	272–480	116–224	{β-VAE, reconstruction based}

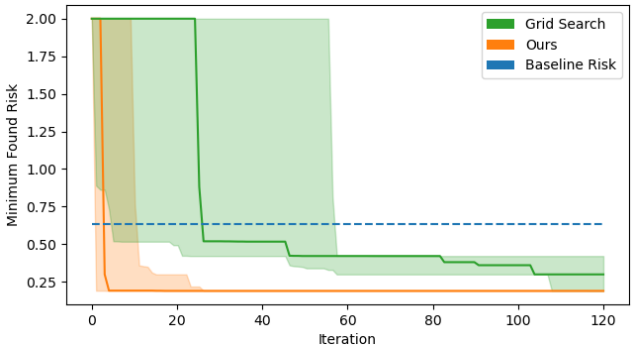


Fig. 5: Risk versus number of iterations for our Bayesian optimizer vs. grid search. The solid line corresponds to the median lowest risk at that epoch and the shaded region shows the 75^{th} to 25^{th} percentile evaluated over 100 trials with different random seeds.

picking a random point in the search space and sweeping all parameters (increments of 8 px. for OOD detector size and 32 px. for YOLO input size). Our modified Bayesian optimization converges faster than grid search, finding the minimum solution within 30 iteration 75% of the time.

The minimum risk and corresponding configuration are shown in Table IV (Combined @4Hz). We note that in all experiments the β-VAE OOD detector outperformed the reconstruction based OOD detector in terms of risk due to the reconstruction based OOD detector's high probability of deadline misses. We also note that the YOLO parameters for the minimum risk configuration are different than baseline case, indicating the importance of co-design. Fig. 6 shows a visualization of the risk surface for the design of YOLO based AEBS with OOD detector at input frequency 4 Hz. We observe that at the selected thresholds, small input images for the OOD detector and YOLO tend to dramatically increase risk (Fig. 6, upper left). This makes sense as in this minimum risk configuration, a majority of the remaining risk is supplied by $P(E_1)$ (Fig. 6, bottom right). As the severity of $P(E_0)$ causes it to contribute more to the overall risk, a configuration was selected during the minimization where $P(E_0)$ is low compared to $P(E_1)$ and relatively invariant with respect to other design parameters (Fig. 6, upper right). In Fig. 6 (bottom left) we also see $P(E_e)$, the probability of YOLO missing a deadline, across design parameters. As expected, we see that this increases for larger input sizes, but note that it is not the dominant factor in our risk plot. This makes sense as deadline misses contribute to E_0, which is small at the optimal solution compared to the contribution of E_1, where more deadline misses can reduce the FPR.

Finally, we perform the risk minimization again, but assume a deadline of 200 ms (5 Hz) to see if our design methodology will allow us to achieve a less risky solution that can sustain a higher sampling rate and is therefore applicable at higher vehicle speeds than the baseline (Table IV: Combined @5Hz). Unfortunately there are no feasible solutions at this rate given our embedded hardware. We also reran the optimization with deadlines every 333 ms (3 Hz) to see if the extra execution time would allow us to further reduce risk at the expense of vehicle speed (Table IV: Combined @3Hz). In this case average utilization is decreased well below the

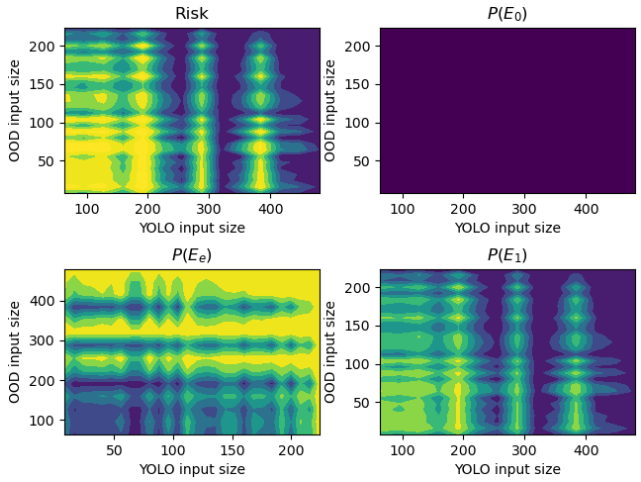


Fig. 6: Projection of risk, $P(E_e)$, $P(E_0)$, and $P(E_1)$ onto OOD and YOLO design parameters. All parameters not specified in a plot are set to the values that produce the minimum risk. Upper left: impact of input sizes on risk. Lower left: impact input sizes on $P(E_e)$. Upper right: impact of input sizes on $P(E_0)$. Lower right, impact of input sizes on $P(E_1)$.

baseline due to the extra available time, but the overall minimum risk is on par with the 4 Hz case. In this case the functional performance of the OOD detector and YOLO object detector limits the overall risk. This indicates that while meeting deadlines is critical to finding a minimal risk solution, setting arbitrarily long deadlines (even if vehicle's speed is slow enough to allow it) does not necessarily help reduce risk further.

VI. LIMITATIONS

While this work shows the utility of an OOD detector as a safety monitor and the advantages of using risk in the co-design of LECs, there are still challenges that must be addressed to bridge the gap between a simulated case study and real transportation systems. Most importantly, there is an implicit assumption that the datasets used for training and validation incorporate the same distribution of edge cases that the system will experience during operation. If this assumption is not met, the risk returned by the co-design is not valid and guaranteeing that this assumption is met may not be feasible for real-world datasets. In this paper a simulated dataset was used to help ensure sufficient coverage of scenarios and reduce the time required to collect data. However, for a real dataset, collecting edge case scenarios and OOD samples could prove dangerous or costly and simulation of such scenarios does not guarantee that the analysis is valid for the corresponding physical system.

Furthermore, we assume the results of the OOD detector and EC are independent across time when determining the probabilities used in the risk analysis. In a practical system, such an assumption is not feasible as previous control actions affect future samples. Also, environmental conditions that determine if a sample is OOD are unlikely to change much between consecutive samples. Incorporating these effects into the risk analysis is a future area of research.

Additionally, this work did not include a study of multimodal AEBSs. Multimodal sensor data is common in

robotic and transportation systems and, in principle, such a system can still be modeled as a binary classifier or ensemble of binary classifiers. However, given a specific system architecture, multimodal input data may allow additional architectural enhancements that can help reduce risk.

VII. CONCLUSION

We addressed the problem of co-designing an OOD detector and LEC for use in an AEBS. While previous works have only focused on reducing the execution times of individual components or increasing their accuracy, our experiments show that the tradeoff between the functional and non-functional performance of each component needs to be taken into consideration when designing for safety. We observed that with our design methodology we were able to reduce risk below that of a baseline system while maintaining the same resource utilization, but that a design approach where both components are developed independently may not yield such a solution. We also demonstrated that our design methodology reduces the time to find a minimal risk solution. This work shows that deploying OOD detectors as safety monitors is feasible, but must be done as part of a co-design process to prevent inadvertently increasing risk.

REFERENCES

[1] F. Cai and X. Koutsoukos, 'Real-time Out-of-distribution Detection in Learning-Enabled Cyber-Physical Systems," in *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, Apr. 2020, pp. 174–183, doi: 10.1109/ICCPS48487.2020.00024.

[2] M. Yuhas and A. Easwaran, "Demo Abstract: Real-Time Out-of-Distribution Detection on a Mobile Robot," in *RTSS@Work 2022*, Dec. 2022, pp. 26–28.

[3] L. Alecu *et. al.*, "Can we reconcile safety objectives with machine learning performances?" presented at Embedded Real Time Systems 2022, Toulouse, France, Jun. 2022.

[4] Y. -C. Hsu, Y. Shen, H. Jin, and Z. Kira, "Generalized ODIN: Detecting Out-of-Distribution Image Without Learning From Out-of-Distribution Data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020, pp. 10951–10961, doi: 10.1109/CVPR42600.2020.01096.

[5] J. An and S. Cho. (2015). Variational Autoencoder based Anomaly Detection using Reconstruction Probability [PDF]. Available: http://dm.snu.ac.kr/static/docs/TR/SNUDM-TR-2015-03.pdf.

[6] A. Vasilev *et. al.*, "q-Space Novelty Detection with Variational Autoencoders," in *Computational Diffusion MRI*, Oct. 2019, pp. 113–124, doi: 10.1007/978-3-030-52893-5.

[7] S. Ramakrishna, Z. Rahiminasab, G. Karsai, A. Easwaran, and A. Dubey, "Efficient Out-of-Distribution Detection Using Latent Space of β-VAE for Cyber-Physical Systems," *ACM Transactions on Cyber-Physical Systems*, vol. 6, no. 2, pp. 15:1–15:34, Apr. 2022, doi: 10.1145/3491243.

[8] M. Yuhas, D. J. X. Ng, and A. Easwaran, "Design Methodology for Deep Out-of-Distribution Detectors in Real-Time Cyber-Physical Systems," in *2022 IEEE 28th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, Aug. 2022, pp. 180–185, doi: 10.1109/RTCSA55878.2022.00025.

[9] C. Hartsell, S. Ramakrishna, A. Dubey, D. Stojcsics, N. Mahadevan, and G. Karsai, "ReSonAte: A Runtime Risk Assessment Framework for Autonomous Systems," in *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, May 2021, pp. 118–129, doi: 10.1109/SEAMS51251.2021.00025.

[10] E. Bekiaris and A. Stevens, "Common risk assessment methodology for advanced driver assistance systems," *Transport Reviews*, vol. 25, no. 3, pp. 283–292, Feb. 2005, doi: 10.1080/0144164042000335797.

[11] C. Becker, J. Brewer, and L. Yount, "Safety of the Intended Functionality of Lane-Centering and Lane-Changing Maneuvers of a Generic Level 3 Highway Chauffeur System," NHTSA, Washington, DC, USA, Rep. DOT-VNTSC-NHTSA-19-02, 2020.

[12] L. Sun, Y. -F. Li, and E. Zio, "Comparison of the HAZOP, FMEA, FRAM, and STPA Methods for the Hazard Analysis of Automatic Emergency Brake Systems," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems Part B: Mechanical Engineering*, vol. 8, no. 3, pp. 031104-1–031104-14, Oct. 2021, doi: 10.1115/1.4051940.

[13] W. Kang, K. Lee, J. Lee, I. Shin, and H. S. Chwa, "LaLaRAND: Flexible Layer-by-Layer CPU/GPU Scheduling for Real-Time DNN Tasks," in *2021 IEEE Real-Time Systems Symposium (RTSS)*, Dec. 2021, pp. 329–341, doi: 10.1109/RTSS52674.2021.00038.

[14] N. Ling, X. Huang, Z. Zhao, N. Guan, Z. Yan, and G. Xing, "BlastNet: Exploiting Duo-Blocks for Cross-Processor Real-Time DNN Inference," in *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, Nov. 2022, pp. 91—105, doi: 10.1145/3560905.3568520.

[15] R. Salay, R. Queiroz, and K. Czarnecki, "An analysis of ISO 26262: Using machine learning safely in automotive software," Sep. 2017, *arXiv:1709.02435*.

[16] S. J. Reddi, S. Kale, and S. Kumar, "On the Convergence of Adam and Beyond," in *6th International Conference on Learning Representations, ICLR 2018*, Apr. 2018, pp. 1–23.

[17] J. Chu, T. Zhao, J. Jiao, Y. Yuan, and Y. Jing, "SOTIF-Oriented Perception Evaluation Method for Forward Obstacle Detection of Autonomous Vehicles," in *IEEE Systems Journal*, doi: 10.1109/JSYST.2023.3234200.

[18] W. S. Lee, D. L. Grosh, F. A., Tillman, and C. H. Lie, "Fault Tree Analysis, Methods, and Applications – A Review," *IEEE Transactions on Reliability*, vol. R–34, no. 3, pp. 194–203, Aug. 1985, doi: 10.1109/TR.1985.5222114.

[19] T. Abdelzaher *et al.*, "Scheduling IDK classifiers with arbitrary dependences to minimize the expected time to successful classification," *Real-Time Systems*, Mar. 2023, doi: 10.1007/s11241-023-09395-0.

[20] P. Pazzaglia, C. Mandrioli, M. Maggio, and A. Cervin, "DMAC: Deadline-miss-aware control," in *31st Euromicro Conference on Real-Time Systems (ECRTS 2019)*, Jul. 2019, pp. 1:1–1:24, doi: 10.4230/LIPIcs.ECRTS.2019.1.

[21] N. Vreman, C. Mandrioli, and A. Cervin, "Deadline-Miss-Adaptive Controller Implementation for Real-Time Control Systems," in *2022 IEEE 28th Real-Time and Embedded Technology and Application Symposium (RTAS)*, May 2022, pp. 13–26, doi: 10.1109/RTAS54340.2022.00010.

[22] C. -Y. Wang, A. Bochkovskiy, and H. -Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," Jul. 2022, *arXiv:2207.02696*.

[23] M. Malu, G. Dasarathy, and A. Spanias, "Bayesian Optimization in High-Dimensional Spaces: A Brief Survey," in *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*, Jul. 2021, pp. 1–8, doi: 10.1109/IISA52424.2021.9555522.

[24] S. Wang and S. H. Ng, "Partition-Based Bayesian Optimization for Stochastic Simulations," in *2020 Winter Simulation Conference (WSC)*, Dec. 2020, pp. 2832–2843, doi: 10.1109/WSC48552.2020.9384014.

[25] J. Hu, Y. Jiang, J. Li, and T. Yuan, "Alternative Acquisition Functions of Bayesian Optimization in Terms of Noisy Observation," in *Proceedings of the 2021 European Symposium on Software Engineering*, Aug. 2021, pp. 112–119, doi: 10.1145/3501774.3501791.

[26] D. Nguyen, S. Gupta, S. Rana, A. Shilton, and S. Venkatesh, "Bayesian Optimization for Categorical and Category-Specific Continuous Inputs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Feb. 2020, pp. 5256–5263.

[27] Z. Wu, F. Li, Y. Zhu, K. Lu, an M. Wu, "Design of a Robust System Architecture for Tracking Vehicle on Highway Based on Monocular Camera," *Sensors*, vol. 22, no. 9, May 2022, doi: 10.3390/s22093359.

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, Nov. 2017, pp. 1–16.

[29] R. Padilla, W. L. Passos, T. L. B. Thadeu, S. L. Netto, and E. A. B. da Silva, "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," *Electronics*, vol. 10, no. 3, Feb. 2021, doi: 10.3390/electronics10030279.

[30] *MUTLI-PROCESS SERVICE*, vR520, NVIDIA, Santa Clara, CA, USA, Oct. 2022, pp. 1–37.

[31] J. Krampe and M. Junge, "Injury Severity for Hazard & Risk Analyses: Calculation of ISO 26262 S-parameter Values from Real-World Crash Data," *Accident Analysis & Prevention*, vol. 138, p. 105321, Apr. 2020, doi:10.1016/j.aap.2019.105321.