Received: 7 April 2022 | Revised: 15 June 2022 | Accepted: 29 July 2022

IET Cyber-Physical Systems: Theory & Applications

DOI: 10.1049/cps2.12034

ORIGINAL RESEARCH

Real-time out-of-distribution detection in cyber-physical systems with learning-enabled components

Feiyang Cai | **Xenofon Koutsoukos**

Institute for Software Integrated Systems, Vanderbilt University, Nashville, Tennessee, USA

Correspondence

Feiyang Cai, Institute for Software Integrated Systems, Vanderbilt University, 1025 16th Ave S, Nashville, TN 37212, USA.
Email: feiyang.cai@vanderbilt.edu

Funding information

National Science Foundation, Grant/Award Number: CNS 1739328; Defense Advanced Research Projects Agency, Grant/Award Number: FA8750-18-C-0089

Abstract

Learning-enabled components (LECs) such as deep neural networks are used increasingly in cyber-physical systems (CPS) since they can handle the uncertainty and variability of the environment and increase the level of autonomy. LECs, however, may compromise system safety since their predictions may have large errors, for example, when the data available at runtime are different than the data used for training. This study considers the problem of efficient and robust out-of-distribution detection for learning-enabled CPS. Out-of-distribution detection using a single input example is typically not robust and may result in a large number of false alarms. The proposed approach utilises neural network architectures that are used to compute efficiently the nonconformity of new inputs relative to the training data. Specifically, variational autoencoder and deep support vector data description networks are used to learn models for the real-time detection of out-of-distribution high-dimensional inputs. Robustness can be improved by incorporating saliency maps that identify parts of the input contributing most to the LEC predictions. We demonstrate the approach using simulation case studies of an advanced emergency braking system and a self-driving end-to-end controller, as well as a real-world data set for autonomous driving. The experimental results show a small detection delay with a very small number of false alarms while the execution time is comparable to the execution time of the original LECs.

KEYWORDS

anomaly detection, inductive conformal prediction, out-of-distribution, saliency maps, self-driving vehicles

1 | INTRODUCTION

Learning-enabled components (LECs) such as neural networks are used in many classes of cyber-physical systems (CPS). Semi-autonomous and autonomous vehicles, in particular, are CPS where LECs can play a significant role in perception, planning, and control if they are complemented with methods for analysing and ensuring safety [11, 33]. However, there are characteristics of LECs that can complicate safety analysis. LECs encode knowledge in a form that is not transparent. Deep neural networks (DNNs), for example, capture features in a multitude of activation functions that cannot be inspected to ensure that the LEC operates as intended. High levels of autonomy require high-capacity models that further obscure the system operation. Even if an LEC is trained and tested extensively, it is typically characterised by a non-zero error rate. More importantly, the error estimated at design time may be different than the true error because of out-of-distribution data.

Since training data sets are necessarily incomplete, safety assessment at design time is also incomplete. Design-time verification and analysis methods must be combined with runtime monitoring techniques that can be used for safety assurance. In real-world CPS, the uncertainty and variability of the environment may result in data that are not similar to the data used for training. Although models such as DNNs generalise well if the training and testing data are sampled from the same distribution, out-of-distribution data may lead to

large errors. An LEC is trained and tested using data available at design time but must be deployed in a real system and operate under possibly different conditions. Testing ensures that the error is satisfactory for a large number of examples. However, during the system operation, the LEC may still encounter out-of-distribution inputs. Out-of-distribution detection must quantify how different are the new test data are from the training data and raise an alarm to indicate that the LEC may give a prediction with a large error. Detection methods must be robust and limit the number of false alarms while being computational efficient for real-time monitoring.

The inputs to the LECs in CPS are normally high-dimensional measurements from sensors, such as cameras, LIDAR, and RADAR. It is difficult and not intuitive to define and understand the out-of-distribution data in the feature space of the original high-dimensional data. Out-of-distribution data are typically due to the different conditions and environments in which the data are collected. Such scenarios and conditions are configured using a series of latent variables in a low-dimensional space. Therefore, the out-of-distribution data considered in this study can be defined as the data whose configurable latent variables are not within the distribution of the training dataset. Although changes or deviations from the training distribution in some parameters cannot be detected because only trivial changes have occurred in the data space. Our detection task is to quantify the change in the data space caused by these configurable parameters and to alert when this change exceeds a certain threshold.

The proposed approach is based on conformal prediction (CP) [4, 35] and conformal anomaly detection (CAD) [19]. The main idea of these methods is to test if a new input example conforms to the training data set by utilising a *nonconformity measure* which assigns a numerical score indicating how different the input example is from the training data set. The next step is to define a p-value as the fraction of observations that have nonconformity scores greater than or equal to the nonconformity scores of the training examples, which is then used for estimating the confidence of the prediction for the test input. In order to use the approach online, inductive conformal anomaly detection (ICAD) is introduced in ref. [20], where the original training set is split into the proper training set and the calibration set, and the p-values are computed relative to calibration examples. If a p-value is smaller than a predefined anomaly threshold ϵ, the test example can be classified as an anomaly. The approach is used for sequential anomaly detection of time trajectories in ref. [19] by using the nearest neighbours and Hausdorff distance measuring the nonconformity between trajectories. Combined with exchangeability martingales, ICAD is used for change-point detection in ref. [34].

However, there are still challenges in applying such methods for real-time detection of high-dimensional inputs in CPS. Existing methods rely on nonconformity measures computed using k-nearest neighbours and kernel density estimation and cannot scale to LECs with high-dimensional inputs. Out-of-distribution detection using a single example is typically not robust and may result in a large number of false alarms that inhibit the CPS operation. Methods based on martingales that incorporate multiple examples are not applicable directly to CPS because the input sequence is time-correlated and not exchangeable.

The main contribution of the study is an approach for real-time detection of out-of-distribution inputs. The approach leverages inductive conformal prediction and anomaly detection. In order to handle high-dimensional inputs in real-time, variational autoencoders (VAEs) [18] and deep support vector data description (SVDD) [29] are utilised for efficient computation of the nonconformity score, which enables the real-time detection of out-of-distribution high-dimensional inputs. The VAE or the SVDD is trained using the same training dataset as the monitored LEC. The learning model can work properly when the test data are from the same distribution as the training dataset, but does not generalise well to the out-of-distribution data. By testing whether the learning model is behaving normally, we can detect whether the test data is sampled from the same distribution as training dataset. Specifically, VAE model should accurately reconstruct the in-distribution input but reconstruct the out-of-distribution data with a low quality; SVDD model should map the in-distribution data as close to the centre of the hypersphere as possible, but map the out-of-distribution data far away from the centre. Therefore, the reconstruction error of the test example in VAE model and the distance of representation of the test example to the centre of the hypersphere in the SVDD model can be used as the nonconformity measure evaluating the difference between the test example and the training dataset. Furthermore, in order to use multiple examples for detection, we apply different heuristic techniques for VAE- and SVDD-based methods. VAE is a generative model that allows generating multiple examples in real-time similar to the input and computing multiple p-values that increase the robustness of detection. SVDD is a model trained to perform anomaly detection, and in our method, it is combined with a test based on a sliding window. It should be noted that the VAE and SVDD neural networks may exhibit an intrinsic error of computing nonconformity scores. However, the robustness of the detection is improved considerably by taking into account multiple input examples and comparing them with the calibration nonconformity scores.

Another contribution of the study is a method for improving the out-of-distribution detection by incorporating saliency maps. Saliency maps aim to identify parts of the input that contribute most to the LEC predictions [31, 37]. In our approach, a saliency map is computed to quantify how much the input features contribute to the LEC output and then is used to weight the contribution of the input features to the nonconformity scores. Therefore, the detection algorithm weights the input features based on their influence on the output of the LEC. The main benefit of this method is to decrease the impact of nonconformal input features that do not contribute to the LEC prediction. For high-dimensional inputs such as images, for example, it is possible that parts of the image do not affect the LEC output. As an illustrative example, the VAE may have difficulty generating fine-

granularity details of the original input image, however, such fine-granularity details may not affect the LEC output. We integrate two algorithms for computing saliency maps into the approach: (1) intergrated-gradients optimised saliency (I-GOS) [27] and (2) VisualBackProp (VBP) [6]. The algorithms are very efficient and can be used in real-time.

The final contribution is the empirical evaluation using (1) an advanced emergency braking system (AEBS), (2) a self-driving end-to-end controller (SDEC), and (3) an autonomous vehicle seasonal dataset (AVSD). The first two case studies are implemented in CARLA [10], an open-source simulator for self-driving cars. AVSD evaluates the approach using the Ford autonomous vehicle seasonal data set [1]. The AEBS uses a perception LEC to detect the nearest front obstacle on the road and estimate the distance from the host vehicle based on camera images. The distance together with the velocity of the host car are used as inputs to a reinforcement learning controller whose objective is to comfortably stop the vehicle. Out-of-distribution inputs are generated by varying a precipitation parameter provided by CARLA, which introduces visual effects that may cause large error in the distance estimation resulting in a collision. The simulation results demonstrate a very small number of false positives and a detection delay less than 1 s. For the SDEC that comes with CARLA [10], the empirical evaluation shows that the proposed method can be used to detect a class of physically realisable attacks in end-to-end autonomous driving presented in ref. [8]. The attacks are realised by painted lines on the road to cause the self-driving car to follow a target path. The objective of the AVSD is to estimate the heading changes of a host vehicle from images captured by a camera. A neural network is trained in specific weather conditions and traffic scenarios but may encounter different ones. The evaluation results show that the proposed approach is effective using a real-world data set. For all cases, the execution time of the detection method is comparable to the execution time of the original LECs, which demonstrates that the method can be used in real-time.

The rest of the study is organised as follows. Section 2 discusses the related work. Section 3 describes the system model and the problem. Sections 4 and 5 present the detection algorithms using VAE and SVDD as the underlying neural network architecture respectively. Section 6 extends the algorithms by incorporating the saliency maps. Section 7 presents the evaluation results and Section 8 concludes the study.

2 | RELATED WORK

Verification and assurance of CPS with machine learning components is considered in ref. [30] in a broader context of verified artificial intelligence. The challenges discussed in ref. [30] include the integration of design-time and runtime methods to address the undecidability of verification in complex systems and environment modelling. Out-of-distribution detection can be used with recovery and reconfiguration techniques to complement design-time verification. Focussing on design-time techniques, an approach to identify regions of the input space that lack training data and potentially larger errors is presented in ref. [15]. The approach could be adapted to predict at runtime if new inputs are from regions covered during training or not. Compositional falsification of CPS with machine learning components is introduced in ref. [11] and demonstrated with a simulated AEBS. The approach is applied at design time for identifying executions that falsify temporal logic specifications and also identifies regions of uncertainty where additional analysis and runtime monitoring is required. A related approach for simulation-based adversarial test generation for autonomous vehicles with machine learning components is presented in ref. [33]. The technique is also used at design time to increase the reliability of autonomous CPS and can provide additional training data for out-of-distribution detection.

Detection of out-of-distribution examples in neural networks has received considerable attention, especially in the context of classification tasks in computer vision [16]. Correctly classified examples tend to have greater maximum softmax probabilities than erroneously classified and out-of-distribution examples. An approach for improving detection by training anomaly detectors is proposed in [17]. An approach for reducing the number of false alarms of out-of-distribution image detection by adding small perturbations to the input is presented in ref. [21]. The idea is similar to randomly sampling from the latent space of the VAE that can also improve the reliability of the detection. Such detection techniques do not take into consideration the dynamical behaviour of CPS and can exhibit a large number of false alarms. An approach that aims to detect novelties based on the reconstruction error of an autoencoder for a single input is presented in ref. [28]. The approach is used for safe visual and LIDAR-based navigation of mobile robots. A similar approach using VAEs is proposed in ref. [23] to estimate the uncertainty for a collision prediction task for a robot car. As discussed in refs. [23, 28], out-of-distribution detection and, in general, uncertainty estimation is an important research direction for providing a more robust detection.

Conformal anomaly detection (CAD) is proposed in ref. [19] based on conformal prediction (CP) [35]. The essential element in these methods is the definition of a nonconformity measure defined by a function A that measures how different a test example from the training data set. Then, a p-value is defined as the fraction of observations that have nonconformity scores greater than or equal to the nonconformity scores of the training examples. A small p-value corresponds to a strange example relative to the training data set. The efficiency of CAD method can be improved by using inductive conformal anomaly detection (ICAD) [20], where the original training set is split into a proper training set and a calibration set, and the p-values are computed relative to calibration examples. Several anomaly detection approaches based on the CAD and ICAD are raised in the literature, but the nonconformity measures are defined differently, such as k-nearest neighbour (k-NN) [20], kernel density estimation (KDE) [32], and sub-sequence local outlier factor [20] nonconformity measures. Moreover, ICAD is combined with

exchangeability martingales in ref. [34] for change-point detection. If the test input sequence is exchangeable, or it is invariant with respect to any random permutation of inputs, the corresponding p-values should be independent and uniformly distributed between 0 and 1. The martingale can be used to test if p-values are independent and uniformly sampled from 0 to 1, and furthermore, if the input sequence is exchangeable.

Out-of-distribution detection may be oversensitive and raise alarms when parts of the input that do not affect the LEC output are changed slightly. Saliency maps are a visualisation tool for identifying which parts of the input contribute most to the LEC predictions and are used in refs. [25, 36] to help to understand if the models focus on reasonable cues in an input image. Saliency maps can be computed using gradient-based methods [31] and deconvolution [37]. The former approach uses backpropagation to compute the partial derivatives with respect to the pixels in the input image, and the latter uses deconvolution to map the feature activities in intermediate layers to the input pixel space. Layer-wise relevance propagation (LRP) is introduced in ref. [3] as a method to compute partial prediction contributions for input representations by propagating the prediction back until the input layer using the network weights and the neural activations created by the forward pass. An integrated-gradients optimised saliency (I-GOS) algorithm is raised in ref. [27] whose basic idea is to optimise the saliency map so that the classification score on the saliency-masked image would maximally decrease. Therefore, the saliency map indicates the influence of the input features on the prediction. VisualBackProp [6] is developed initially as a debugging tool and computes the saliency maps by backpropagating the information of the feature maps from deeper layers while simultaneously increasing the resolution.

Adversarial examples can deceive the neural network into predicting erroneous outputs by crafting the inputs with deliberately designed perturbations [14]. Moreover, adversarial attacks have been implemented to the physical domain trying to design adversarial examples that are physically realisable [12]. Recently, such physical adversarial examples are deployed directly into an autonomous driving system, a typical example of cyber-physical systems [8]. Although several defences are developed to withstand such adversarial attacks [24], it has been proven that the defences can be bypassed by adaptive attacks whose objective is to fool the neural network and the defence technique simultaneously [9]. The physically realisable adaptive attacks are underexplored in the literature.

3 | SYSTEM MODEL AND PROBLEM FORMULATION

CPSs use extensively LECs to perform various tasks in order to increase the level of autonomy. A typical simplified CPS architecture with LECs (e.g., DNNs) for perception and control is shown in Figure 1. A perception component observes and interprets the environment and provides information to a controller, which, possibly using additional sensors (feedback from the plant), applies an action to the plant in order to achieve some task. In response to this action, the state of the physical plant changes and the environment must be observed and interpreted again to continue the system operation. An end-to-end control architecture from perception to actuation can also be used.

An LEC is designed using learning methods, such as supervised and reinforcement learning. We assume that the LECs are successfully trained, and the training and testing errors are satisfactory. However, the training and testing data sets at design time are necessarily incomplete and may underrepresent safety-critical cases. Out-of-distribution inputs, in particular, which have not been used for training or testing, may lead to large errors and compromise safety.

The inputs to perception and end-to-end control LECs are high-dimensional measurements from sensors such as cameras, LIDAR, and RADAR. Out-of-distribution data in CPS typically due to data being collected under different conditions or in a different environment. The scenarios and conditions under which the dataset was generated can be configured using a series of latent variables in a low-dimensional space. Therefore, the out-of-distribution can be defined as the data whose latent variables are from a different distribution of the training dataset.

The problem considered in this study is robustly detecting out-of-distribution inputs in real-time. Out-of-distribution detection is crucial, for example, in order to enable decision making by switching to a different control architecture or human supervision. During the system operation, the inputs arrive one by one to the perception LEC. After receiving each input, the objective is to compute a valid measure of the difference between the test input and training dataset in the data space and to alert when this difference exceeds a certain threshold.

Online detection algorithms must be robust with a small number of false alarms. The detection algorithms using a single example are typically not robust and may result in a large number of false alarms. The inputs in CPS are time-correlated, and therefore, they are not independent, which imposes a

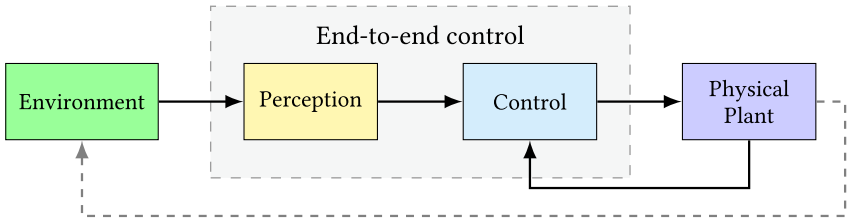


FIGURE 1 Simplified CPS control architecture.

significant challenge to use multiple examples in detection. Furthermore, for learning-enabled CPS, out-of-distribution detection must be performed in real-time, which is very challenging because inputs to the LECs are high-dimensional measurements. The time and memory requirements of the detection must be similar to the requirements of the LECs used in the CPS architecture. Typical nonconformity measures such as the k-nearest neighbour (k-NN) nonconformity measure [20] and the kernel density estimation (KDE) nonconformity measure [32] cannot scale to high-dimensional inputs because they require either storing the training data set or estimating the density in a high-dimensional space. It should be noted that the system state can also be the input to the LEC. How to estimate the state and how large the estimation error is are beyond the scope of the problem considered in this study.

4 | VAE-BASED DETECTION

Variational autoencoder (VAE) is a generative model that learns parameters of a probability distribution to represent the data [18]. A VAE consists of an encoder, a decoder, and a loss function. The objective is to model the relationship between the observation x and the low-dimensional latent variable z using the loss function $\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - D_{\mathrm{KL}}[q_\phi(z|x)\|p(z)]$, where θ and ϕ are neural network parameters. The first term in the loss function is the model fit, and the second is the KL divergence between the approximate posterior and the prior of z. A popular choice for the prior is the Gaussian distribution. VAE-based methods can utilise the reconstruction error or reconstruction accuracy for anomaly detection [2]. In this section, we introduce a VAE-based detection method based on inductive conformal anomaly detection. The VAE allows generating multiple examples that are similar to the input. If these examples are not conformal to the training data, many of the corresponding nonconformity scores will be very large indicating the test example is out of the distribution of the training data set.

4.1 | Offline training and nonconformity measure

Let us consider an LEC $y = f(x)$ defining a mapping from input x to output y. It should be noted that in this work only the inputs are taken into consideration for out-of-distribution detection. VAE is employed to measure the nonconformity between the test input and the input distribution of the training dataset of the LEC. Besides, VAE is an unsupervised learning method in the sense that the labels are not required in addition to the data inputs. Therefore, the training dataset of the monitored LEC excluding the labels is used to train the VAE. The set of input examples used for training the VAE is denoted by $\{x_1, \ldots, x_l\}$. During the system operation, a sequence of inputs denoted by $\{x'_1, \ldots, x'_t, \ldots\}$ is processed one-by-one. The task of the out-of-distribution detection is to quantify how different the input sequence is from the training data set. If the difference is large, the algorithm raises an alarm indicating that the LEC may generate an output y with a large error compared to the testing error obtained at design time.

Since the observations of the environment are highly time-correlated, the collected training data are not exchangeable. As suggested in [35], reshuffling—a random permutation for the training data set—is performed before training the VAE. After reshuffling, the training data set is split into a proper training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ and a calibration set $\{(x_{m+1}, y_{m+1}), \ldots, (x_l, y_l)\}$. For each example in the calibration set, a function A is used to compute the nonconformity measure that assigns a numerical score indicating how different a test example is from the training data set. The nonconformity scores of the calibration examples are sorted and stored in order to be used at runtime.

Given a new input x'_t, the nonconformity score α'_t is computed using the nonconformity function A relative to the proper training set, $\alpha'_t = A\left(\{x_1, \ldots, x_m\}, x'_t\right)$. The computation requires evaluating the nonconformity (strangeness) of x'_t relative to $(x_1, \ldots, x_m)$. The choice of the nonconformity function A must ensure computing informative nonconformity scores in real-time. Using k-NN, for example, requires storing the training data set, which is infeasible for high-dimensional inputs. Instead, we learn an appropriate neural network architecture that is trained offline using the proper training set and encodes the required information in its parameters. This neural network *monitors* the inputs to the perception or end-to-end control LEC and is used to compute the nonconformity measure in real-time.

Similar to the standard autoencoder, VAE attempts to reconstruct the input image as accurately as possible. For an in-distribution input x, it should be reconstructed by the VAE with a relatively small reconstruction error. Conversely, if the test image is not sampled from the same distribution of the training dataset, VAE cannot generalise well and will reconstruct the test image with a low quality. The reconstruction error is a good indication of the strangeness of the input relative to the training set, and it is used as the nonconformity measure. Similar to the standard autoencoder, VAE attempts to reconstruct the input image as accurately as possible.

We use the squared error between the input example x and generated output example $\hat{x}$ as the nonconformity measure defined as

$$\alpha = A_{\mathrm{VAE}}(x, \hat{x}) = \|x - \hat{x}\|^2. \tag{1}$$

During the offline phase, for each example $x_j : j \in \{m+1, \ldots, l\}$ in the calibration data set, we sample a single reconstructed input $\hat{x}_j$ from the trained VAE and compute the nonconformity score α_j^{Γ} using Equation (1). The precomputed nonconformity scores of the calibration data are sorted and stored in order to be used at runtime. The steps that are performed offline are summarised in Algorithm 1.

Algorithm 1 VAE-based out-of-distribution detection

Input: Input training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, input sequence $(x'_1, \ldots, x'_t, \ldots)$, number of calibration examples $l - m$, number of examples N generated by the VAE, threshold τ and parameter δ of CUSUM detector
Output: Output Boolean variable $Anom_t$
Offline:
Split the training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ into the proper training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ and calibration set $\{(x_{m+1}, y_{m+1}), \ldots, (x_l, y_l)\}$
Train a VAE using the proper training set
for $j = m + 1$ to l **do**
  Generate $\hat{x}_j$ using the trained VAE
  $\alpha_j^{\Gamma} = A_{\text{VAE}}(x_j, \hat{x}_j)$
end for
$\{\alpha_{m+1}, \ldots, \alpha_l\} = \text{sort}(\{\alpha_{m+1}^{\Gamma}, \ldots, \alpha_l^{\Gamma}\})$
Online:
$S_0 = 0$
for $t = 1, 2, \ldots$ **do**
  for $k = 1$ to N **do**
    Sample $\hat{x}'_{t,k}$ using the trained VAE
    $\alpha'_{t,k} = A_{\text{VAE}}(x'_t, \hat{x}'_{t,k})$
    $p_{t,k} = \frac{|\{i=m+1,\ldots,l\} \mid \alpha_i \geq \alpha'_{t,k}|}{l-m}$
  end for
  $M_t = \int_0^1 \prod_{k=1}^{N} \epsilon p_{t,k}^{\epsilon-1} d\epsilon$
  $S_t = \max(0, S_{t-1} + M_t - \delta)$
  $Anom_t \leftarrow S_t > \tau$
end for

4.2 | Online detection

Given a test input x'_t, the p-value p_t is computed as the fraction of calibration examples that have nonconformity scores greater than or equal to α'_t

$$p_t = \frac{|\{i = m+1, \ldots, l\} \mid \alpha_i \geq \alpha_t|}{l}. \tag{2}$$

It should be noted that the computation of the p-value can be performed efficiently online since it requires storing only the calibration nonconformity scores. If $p_t < \epsilon$, the example x'_t can be classified as an anomaly. Using a single p-value for detecting out-of-distribution examples can lead to an oversensitive detector with a large number of false alarms that inhibit the operation of the CPS. Our objective is to incorporate multiple examples into the detection algorithm and compute a sequence of p-values to test if there are many small p-values indicating an out-of-distribution input. The detail of the online detection procedure is described below, and we also provide the architecture in Figure 5 to better illustrate the detection pipeline.

Given an input example x'_t at time t, the encoder portion of a VAE is used to approximate the posterior distribution of the latent space. Typically, the posterior of the latent space is approximated by a Gaussian distribution, whose mean is μ_t and standard deviation is σ_t. Then, N points $\{z'_{t,1}, \ldots, z'_{t,N}\}$ are sampled from the posterior that are used as input to the decoder portion in order to generate multiple new examples $\{\hat{x}'_{t,1}, \ldots, \hat{x}'_{t,N}\}$. Sampling from the posterior generates encodings $\{z'_{t,1}, \ldots, z'_{t,N}\}$ so that the decoder is exposed to a range of variations of the input example and outputs $\{\hat{x}'_{t,1}, \ldots, \hat{x}'_{t,N}\}$.

For each generated example $\hat{x}'_{t,k}$, $k \in \{1, \ldots, N\}$, the nonconformity score $\alpha'_{t,k}$ is computed as the reconstruction error between the test input x'_t and the generated example using Equation (1). Such nonconformity measure A_{VAE} corresponds to the 'NCM' block in the architecture. Then, the p-value $p_{t,k}$ is computed as the fraction of calibration examples that have nonconformity scores greater than or equal to $\alpha'_{t,k}$ using Equation (2). It should be noted that, although the generated examples $\{\hat{x}'_{t,1}, \ldots, \hat{x}'_{t,N}\}$ satisfy the exchangeability assumption, they are sampled from a Gaussian distribution conditioned by the input and are not sampled from the same distribution as the calibration data set. Therefore, the N p-values $\{p_{t,1}, \ldots, p_{t,N}\}$ are not independent and uniformly distributed in [0, 1]. However, if the input x'_t is sampled from the same distribution as the training data set, the p-values will be large, and they can be used for out-of-distribution detection.

At runtime, for every new input example x'_t received by the perception or end-to-end control LEC at time t, a *power martingale* [13] can be computed based on the sequence of p-values for some ϵ as

$$M_t^{\epsilon} = \prod_{k=1}^{N} \epsilon p_{t,k}^{\epsilon-1},$$

and the *simple mixture martingale* [13] can be defined as

$$M_t = \int_0^1 M_t^{\epsilon} d\epsilon = \int_0^1 \prod_{k=1}^{N} \epsilon p_{t,k}^{\epsilon-1} d\epsilon. \tag{3}$$

Both martingales will grow only if there are many small p-values in $\{p_{t,1}, \ldots, p_{t,N}\}$, which will indicate an out-of-distribution input. If most of the p-values are relatively greater than 0, the martingales are not expected to grow. We use simple mixture martingale in our approach to avoid parameter tuning required for the power martingale. Such computation corresponds to the 'Martingale test' block in Figure 2.

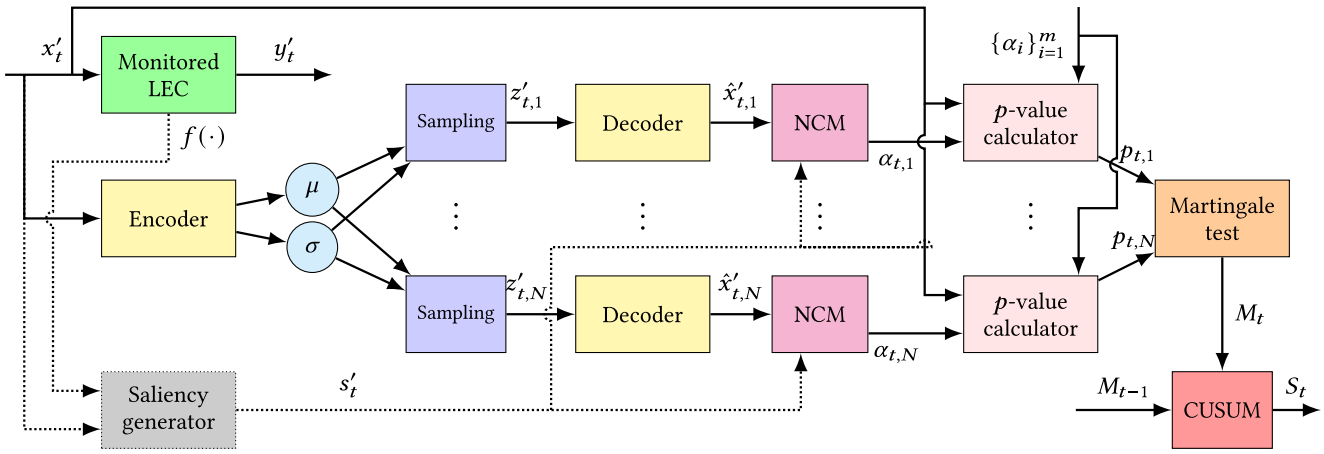


FIGURE 2 Architecture of the VAE-based out-of-distribution detection. The grey block and dotted arrow are not executed when the saliency map is not used in the detection. NCM stands for nonconformity measure, which corresponds to A_{VAE} when the saliency map is not used, but to $A_{\text{VAE-S}}$ when the saliency map is used.

In order to robustly detect when M_t becomes consistently large, we use the cumulative sum (CUSUM) procedure [5]. CUSUM is a nonparametric stateful test and can be used to generate alarms for out-of-distribution inputs by keeping track of the historical information of the martingale values. The detector is defined as $S_0 = 0$ and $S_t = \max(0, S_{t-1} + M_t - \delta)$, where δ prevents S_t from increasing consistently when the inputs are in the same distribution as the training data. An alarm is raised whenever S_t is greater than a threshold $S_t > \tau$, which can be optimised using empirical data [5]. Typically, after an alarm, the test is reset with $S_t = 0$.

Algorithm 1 describes the VAE-based real-time out-of-distribution detection. The nonconformity measure can be computed very efficiently by executing the learnt VAE neural network and generating N new examples. The complexity is comparable to the complexity of the perception or end-to-end LEC that is executed in real-time.

5 | SVDD-BASED DETECTION

VAEs and other autoencoder architectures are trained to perform a task other than anomaly detection, assuming that the reconstruction accuracy will be better for in-distribution examples. Deep support vector data description (SVDD) is an architecture trained to perform anomaly detection [29]. The idea is to train a DNN to map the input data into a hypersphere of minimum volume characterised by centre $\boldsymbol{c}$ and radius R. The input space $\mathcal{X}$ is transformed to a compressed output space $\mathcal{Z}$ while minimising the volume of the hypersphere that encloses most of the input representations. Given a training data set $\{x_1, \ldots, x_n\}$, the *one-class deep SVDD* [29] is based on the loss

$$\min_{\mathcal{W}} \quad \frac{1}{n}\sum_{i=1}^{n} \|\phi(x_i;\mathcal{W}) - \boldsymbol{c}\|^2 + \frac{\lambda}{2}\sum_{\ell=1}^{L} \|\boldsymbol{W}^{\ell}\|_F^2,$$

where $\phi(\cdot\,;\mathcal{W}) : \mathcal{X} \to \mathcal{Z}$ denotes the neural network with L hidden layers and sets of weights $\mathcal{W} = \{\boldsymbol{W}^1, \ldots, \boldsymbol{W}^{\ell}, \ldots, \boldsymbol{W}^{L}\}$, $\boldsymbol{c} \in \mathcal{Z}$ is the centre of the hypersphere, and the last term is a weight regulariser with hyperparameter $\lambda > 0$, where $\|\cdot\|_F$ is the Frobenius norm. One-class deep SVDD learns to map the data as close to centre $\boldsymbol{c}$ as possible by penalising the distance from representations to the centre. The deep SVDD neural network must not have bias terms or bounded activation functions, and the centre $\boldsymbol{c}$ can be selected as the mean of the representations from the initial inference on some training data to avoid trivial solutions that map the input space to a single point [29]. Given a new test example x, the distance of the representation $\phi(x;\mathcal{W}^*)$ to the centre $\boldsymbol{c}$ of the hypersphere reflects how different the test example is from the training data set and can be used as a nonconformity measure. In contrast to VAEs, SVDD is not a generative model and cannot be used to sample multiple examples. In order to use effectively the SVDD architecture in our approach, we use a sliding window containing a sequence of inputs as explained below.

5.1 | Offline training and nonconformity measure

The offline phase of the SVDD-based detection algorithm is similar to the VAE-based algorithm, but the only difference is the nonconformity measure. After reshuffling the training data set and splitting it into proper training data set and calibration data set, a deep SVDD model is trained using the proper training data set. The centre of the hypersphere $\boldsymbol{c}$ is fixed as the mean of the representations from the initial pass on the proper training data. After training, the neural network function $\phi(x, \mathcal{W}^*)$ maps an input example x to a representation close to the centre $\boldsymbol{c}$. In-distribution inputs are likely concentrated in a relatively small area in the output space, while the out-of-distribution inputs will be far away from the centre. The distance of the representation to the centre $\boldsymbol{c}$ of the hypersphere can be used to evaluate the strangeness of the test example relative to the proper training set and is defined as the nonconformity measure

$$\alpha = A_{\text{SVDD}}(x) = \|\phi(x; \mathcal{W}^*) - \boldsymbol{c}\|^2. \tag{4}$$

The SVDD-based method also uses a learnt model to calculate the nonconformity score, and it can be used to compute the nonconformity score in real-time. During the offline phase, the nonconformity scores for the calibration data are precomputed using Equation (4) and sorted in order to be used at runtime. The offline phase for the SVDD-based detector is shown in Algorithm 2.

Algorithm 2 SVDD-based out-of-distribution detection

Input: Input training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, input sequence $(x'_1, \ldots, x'_t, \ldots)$, number of calibration examples $l - m$, sliding window size N, detector threshold τ
Output: Output Boolean variable $Anom_t$
Offline:
Split the training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ into the proper training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ and calibration set $\{(x_{m+1}, y_{m+1}), \ldots, (x_l, y_l)\}$
Train a SVDD model using the proper training set
for $j = m + 1$ to l **do**
    $\alpha_j^{\Gamma} = A_{\text{SVDD}}(x_j)$
end for
$\{\alpha_{m+1}, \ldots, \alpha_l\} = \text{sort}(\{\alpha_{m+1}^{\Gamma}, \ldots, \alpha_l^{\Gamma}\})$
Online:
for $t = 1, 2, \ldots$ **do**
    $\alpha'_t = A_{\text{SVDD}}(x'_t)$
    $p_t = \frac{|\{i=m+1,\ldots,l\} \mid \alpha_i \geq \alpha'_t|}{l-m}$
    $M_t = \int_0^1 \prod_{i=t-N+1}^{t} \epsilon p_i^{\epsilon-1} d\epsilon$
    $Anom_t \leftarrow M_t > \tau$
end for

5.2 | Online detection

In order to improve the robustness of out-of-distribution detection, it is desirable to use a sequence of inputs. However, in contrast to the VAE, SVDD is not a generative model and cannot be used to generate multiple examples similar to the input. In the SVDD-based method, we are using a sliding window of inputs $\{x'_{t-N+1}, \ldots, x'_t\}$. Figure 3 illustrates the detailed online detection process of SVDD-based method, where the block 'input masker' is an identity function in this method. In CPS, the inputs arrive at the perception or end-to-end LEC one-by-one and they are time-correlated, and therefore, the inputs within the sliding window are not independent. In order to apply this method to CPS, the size of the sliding window N should be carefully chosen based on the auto-correlation analysis on the nonconformity scores of the input sequence $\{\alpha'_{t-N+1}, \ldots, \alpha'_t\}$. Within this sliding window, the main factor that differentiates consecutive observations are random disturbances and noise. Although the p-values are not guaranteed to be independent and uniformly distributed, out-of-distribution inputs will still result in small p-values, and the martingale test can be used to identify sequences with many small values. In this case, the simple mixture martingale at time t can be defined as

$$M_t = \int_0^1 M_t^{\epsilon} d\epsilon = \int_0^1 \prod_{i=t-N+1}^{t} \epsilon p_i^{\epsilon-1} d\epsilon. \tag{5}$$

Such martingale will grow only if there are many small p-values in this sliding window indicating out-of-distribution inputs are present in the sequence. It should be noted that the martingale M_t does not depend on the order of the input examples $\{x'_{t-N+1}, \ldots, x'_t\}$. Also, M_t must be initialised for the first steps using, for example, random independent and uniformly distributed p-values. Since we already use a sliding window to compute M_t, we employ a stateless detector based on the value M_t and a predefined threshold τ expressed as $M_t > \tau$.

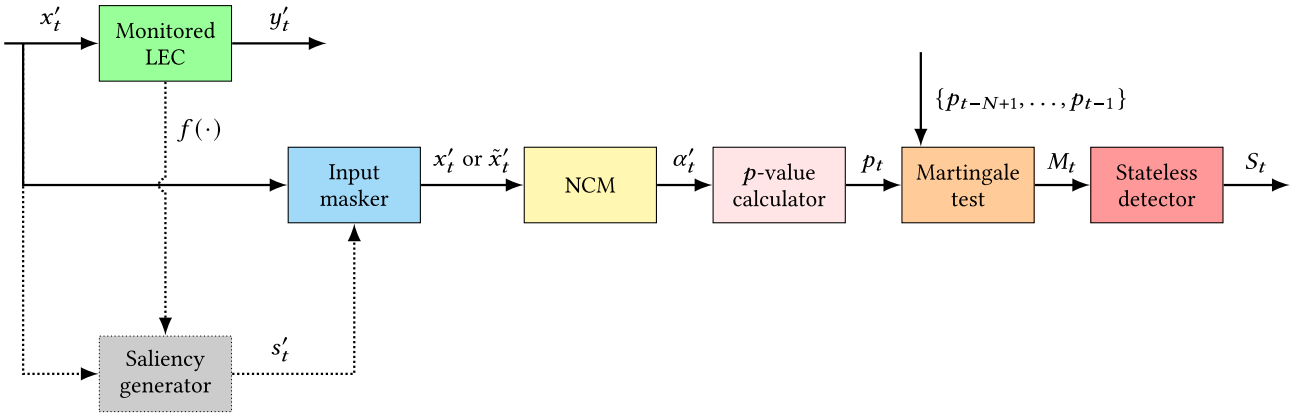


FIGURE 3 Architecture of the SVDD-based out-of-distribution detection. When the saliency map is not used in the detection, the grey block and dotted arrow are not executed, and the input masker is an identity function. NCM stands for nonconformity measure, which corresponds to A_{SVDD}.

Algorithm 2 describes the SVDD-based real-time out-of-distribution detection. Compared with the VAE, the SVDD-based method is more efficient since it does not require generating multiple examples at each step. The martingale M_t can be computed recursively by incorporating the p-value for the new input and omitting the last one in the sliding window.

6 | SALIENCY MAPS

The proposed out-of-distribution detection approach aims to identify inputs that are nonconformal to the training data set. Although the LEC predictions for such inputs may not have large errors, they are generated from a different probability distribution. However, it is possible that some nonconformal features of the input do not contribute to the LEC prediction. For high-dimensional inputs such as camera images, for example, it is possible that parts of the image do not affect the LEC output. As an illustrative example, if the VAE has difficulty generating fine-granularity details of the original input image, the algorithm presented in Section 4 will result in large nonconformity scores. However, such fine-granularity details may not affect the LEC output. This section extends the proposed approach by incorporating *saliency maps* that quantify the spatial support of the LEC prediction of a given input image into the out-of-distribution detection.

The purpose of saliency maps is to identify parts of the input image that contribute most to the LEC predictions [31, 37]. In our approach, a saliency map is computed to quantify how much the input features of the image contribute to the LEC output. Then, the saliency map is used to weight the contribution of the input features to the nonconformity scores. Therefore, the detection algorithm weights the input features based on their influence on the output of the perception or end-to-end LEC in the CPS architecture.

We utilise two algorithms for computing the saliency maps, integrated-gradients optimised saliency (I-GOS) and Visual-BackProp (VBP). I-GOS is introduced in ref. [27] that optimises for the saliency map so that the classification scores on the masked image would maximally decrease, which reflects the input features that have greatest influence on the prediction. VBP is an algorithm that aims to highlight important regions that contribute towards the predictions made by convolutional neural networks [6]. The intuition of VBP is that the feature maps contain less irrelevant information to the prediction when moving deeper into the network. Since the resolution of the feature maps becomes lower for deeper layers, VBP computes the saliency map by back-propagating the information of feature maps from deeper layers while simultaneously increasing the resolution.

Consider an LEC $y = f(x)$ defining a mapping from the input x to output y used to perform regression or classification. The input image x is of size $H \times W \times C$ where H, W, and C denote the height, width, and channel respectively. We denote each input element as $x^{h,w,c}$ where $h \in \{0, \ldots, H-1\}$, $w \in \{0, \ldots, W-1\}$, and $c \in \{0, \ldots, C-1\}$. I-GOS is initially designed for the classification problem. As for regression, the algorithm can be modified by optimising a saliency map s' so that the regression result on the saliency-masked image would maximally change. In order to use the saliency map in the detection algorithm, we convert s' to a greyscale image s of size $H \times W$. VBP uses the feature maps outputted by the convolutional layers and can be used for both regression and classification tasks. In VBP, the generated saliency map s is already greyscaled of size $H \times W$. Moreover, in order to deal with the problem that different images will have different total contributions, the saliency map should be normalised by the sum of contributions for all pixels in the image. In summary, we can define a function $s = G(x_0; f)$ that generates a greyscale saliency map s for the LEC f given input image x_0 using either I-GOS or VBP.

6.1 | VAE-based detection with saliency maps

We define the nonconformity measure by weighting the squared error between the input example x and a generated example $\hat{x}$ from the VAE using the saliency map s

$$\alpha = A_{\text{VAE-S}}(x, \hat{x}, s) = \frac{1}{H \times W \times C} \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} \sum_{c=0}^{C-1} s^{h,w} \times \left(x^{h,w,c} - \hat{x}^{h,w,c}\right)^2. \tag{6}$$

Therefore, the input features that influence the LEC predictions contribute more to the nonconformity measure.

During the offline phase of the algorithm, similar to the VAE-based detection method introduced in Section 4, the training data set is reshuffled and split into a proper training set and a calibration set. The proper training data set is used to train both the LEC and the VAE network. For each example in the calibration data set, we compute the saliency map, and the nonconformity score is computed using Equation (6). The nonconformity scores of the calibration data are sorted and stored for monitoring at runtime.

Algorithm 3 VAE-based out-of-distribution detection

Input: Input training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, input sequence $(x'_1, \ldots, x'_t, \ldots)$, number of calibration examples $l - m$, monitored LEC $f(\cdot)$, number of examples N generated by the VAE, threshold τ and parameter δ of CUSUM detector

Output: Output Boolean variable $Anom_t$

Offline:

1: Split the training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ into the proper training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ and calibration set $\{(x_{m+1}, y_{m+1}), \ldots, (x_l, y_l)\}$

2: Train a VAE using the proper training set

for $j = m + 1$ to l **do**
  Generate $\hat{x}_j$ using the trained VAE
  Compute the saliency map $s_j = G(x_j; f)$
  $\alpha_j^{\Gamma} = A_{\text{VAE-S}}(x_j, \hat{x}_j, s_j)$
end for
$\{\alpha_{m+1}, \ldots, \alpha_l\} = \text{sort}(\{\alpha_{m+1}^{\Gamma}, \ldots, \alpha_l^{\Gamma}\})$
Online:
for $t = 1, 2, \ldots$ **do**
  Compute the saliency map $s'_t = G(x'_t; f)$
  for $k = 1$ to N **do**
    Generate $\hat{x}'_{t,k}$ using the trained VAE
    $\alpha'_{t,k} = A_{\text{VAE-S}}(x'_t, \hat{x}'_{t,k}, s'_t)$
    $p_{t,k} = \frac{|\{i=m+1,\ldots,l\} \mid \alpha_i \geq \alpha'_{t,k}|}{l-m}$
  end for
  $M_t = \int_0^1 \prod_{k=1}^{N} \epsilon p_{t,k}^{\epsilon-1} d\epsilon$
  if $t = 1$ **then**
    $S_t = 0$
  else
    $S_t = \max(0, S_{t-1} + M_{t-1} - \delta)$
  end if
  $Anom_t \leftarrow S_t > \tau$
end for

The detailed online detection procedure is shown in Figure 2. At runtime, given an input example x'_t, the saliency map $s'_t = G(x'_t; f)$ is used to compute the nonconformity score. As described in Section 4, for each input example x'_t, N examples $\{\hat{x}'_{t,1}, \ldots, \hat{x}'_{t,N}\}$ are generated from the VAE. For each generated example $\hat{x}'_{t,k}$, the nonconformity score $\alpha'_{t,k}$ is computed by $\alpha'_{t,k} = A_{\text{VAE-S}}(x'_t, \hat{x}'_{t,k}, s'_t)$ (Equation (6)), which is represented as 'NCM' in the architecture. The corresponding p-value $p_{t,k}$ is calculated as Equation (2). Most of the p-values are expected to be much greater than 0, and the martingale M_t computed by Equation (3) is used to test if x'_t is an out-of-distribution input. Finally, a CUSUM detector is used to generate alarms for out-of-distribution inputs. Algorithm 3 summarises the VAE-based out-of-distribution detection using saliency maps.

6.2 | SVDD-based detection with saliency maps

In order to incorporate the saliency maps into the SVDD-based detection method, the input x is masked by the saliency map s and is used as the new input to the SVDD network. Each element of the masked image denoted by $\tilde{x}$ is computed by $\tilde{x}^{h,w,c} = x^{h,w,c} \cdot s^{h,w}$.

During the offline phase, we compute the saliency map for each example in the training data set and use it to mask the example to create a new training data set $\{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_l, y_l)\}$. Then, the new training data set $\{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_l, y_l)\}$ is reshuffled and split into the proper training set $\{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_m, y_m)\}$ and calibration set $\{(\tilde{x}_{m+1}, y_{m+1}), \ldots, (\tilde{x}_l, y_l)\}$. The proper training set is used to train a new SVDD network which maps the inputs to a representation in a lower-dimensional hypersphere suitable for anomaly detection. The nonconformity measure A_{SVDD} is defined as the distance of the representation to the centre of the hypersphere (Equation (4)). For each example in the new calibration data set, the nonconformity score is computed using A_{SVDD} and sorted for runtime monitoring.

Algorithm 4 SVDD-based out-of-distribution detection using saliency map

Input: Input training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, input sequence $(x'_1, \ldots, x'_t, \ldots)$, number of calibration examples $l - m$, monitored LEC $f(\cdot)$, sliding window size N, detector threshold τ
Output: Output Boolean variable $Anom_t$
Offline:
for $i = 1$ to l **do**
  Compute the saliency map $s_i = G(x_i; f)$
  Mask the input x_i using the saliency map s_i and get $\tilde{x}_i$
end for
Split the training set $\{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_l, y_l)\}$ into the proper training set $\{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_m, y_m)\}$ and calibration set $\{(\tilde{x}_{m+1}, y_{m+1}), \ldots, (\tilde{x}_l, y_l)\}$
Train a SVDD model using the proper training set
for $j = m + 1$ to l **do**
  $\alpha_j = A_{\text{SVDD}}(\tilde{x}_j)$
end for
$\{\alpha_{m+1}, \ldots, \alpha_l\} = \text{sort}(\{\alpha_{m+1}^{\Gamma}, \ldots, \alpha_l^{\Gamma}\})$
Online:
for $t = 1, 2, \ldots$ **do**
  Compute the saliency map $s'_t = G(x'_t; f)$
  Mask the input x'_t using the saliency map s'_t and get input $\tilde{x}'_t$
  $\alpha'_t = A_{\text{SVDD}}(\tilde{x}'_t)$
  $p_t = \frac{|\{i=m+1,\ldots,l\} \mid \alpha_i \geq \alpha'_t|}{l-m}$
  $M_t = \int_0^1 \prod_{i=t-N+1}^{t} \epsilon p_i^{\epsilon-1} d\epsilon$
  $Anom_t \leftarrow M_t > \tau$
end for

During runtime monitoring, given the test input x'_t, the algorithm computes the saliency map $s'_t = G(x'_t; f)$ and uses saliency-masked input $\tilde{x}'_t$ as the input to the SVDD network in order to compute the nonconformity score α'_t as well as the p-value for the test example. Similar to Section 5, a sliding window is used to adapt the martingale test, and a stateless detector is applied to generate alarms for out-of-distribution examples. Such a detection pipeline is illustrated in Figure 3, where the computation of 'Input masker' block corresponds to

the Equation (6). Besides, the SVDD-based detection algorithm using saliency maps is shown in Algorithm 4.

7 | EVALUATION

We evaluate the proposed approach using (1) an advanced emergency braking system (AEBS), (2) a self-driving end-to-end controller (SDEC), and (3) an autonomous vehicle seasonal dataset (AVSD). The AEBS and SDEC are implemented using CARLA [10], an open-source simulator for self-driving cars. AVSD evaluates the approach using the Ford autonomous vehicle seasonal data set [1]. All the experiments are conducted on a 16-core i9 desktop with 32 GB RAM and a single RTX 2080 GPU with 8 GB video memory.

7.1 | Advanced emergency braking system

7.1.1 | Experimental setup

The architecture of the AEBS is shown in Figure 4. A perception LEC is used to detect the nearest front obstacle on the road and estimate the distance. The distance together with the velocity of the host car are used as inputs to a reinforcement learning controller whose objective is to generate the appropriate braking force in order to safely and comfortably stop the vehicle.

The desirable behaviour is illustrated in Figure 5. The AEBS detects a stopped lead car and applies the brake to decelerate and avoid the potential collision. The initial velocity of the host vehicle is v_0, and the initial distance between the host car and the obstacle is d_0. The goal of the controller is to stop the car between $L_{\min}$ and $L_{\max}$. The sampling period used in the simulation is $\Delta t = 1/20$ s. In order to simulate realistic scenarios, we introduce uncertainty into the system. The initial velocity v_0 is uniformly sampled between 90 km/h and 100 km/h, and the initial distance d_0 is approximately 100 m. CARLA allows controlling the precipitation in the simulation using a parameter, which takes values in [0, 100]. For training the perception LEC, and also the VAE and SVDD used for out-of-distribution detection, the precipitation parameter is randomly sampled from the interval [0, 20]. The uncertainty introduced affects the error of the perception LEC. It should be noted that this is just a visual effect, and it does not affect the physical behaviour of the car.

The perception LEC is implemented using a convolutional neural network (CNN), which is trained using supervised learning with a training data set consisting of 8160 images obtained by varying the simulation parameters as described above. The perception LEC has three layers of 24/36/48 × (5 × 5) filters with ReLU activations and 2 × 2 strides, two layers of 64/64 × (3 × 3) filters with ReLU activations and 1 × 1 strides, three fully connected layers of 100/50/10 units with ReLU activations and an output layer of size 1 with Sigmoid activation. After 100-epoch training, the mean absolute errors for training and testing are 0.54 and 0.56 m respectively and are used to select $L_{\min}$ and ensure safety. The reinforcement learning controller is trained using the DDPG algorithm [22] with 1000 episodes and reward function which aims to stop the vehicle between $L_{\min} = 1$ m and $L_{\max} = 3$ m. A simulation run is shown in Figure 6. Initially, the distance between the host and the lead car is 98.02 m, and the velocity of the host car is 97.13 km/h (= 26.98 m/s). After 140 steps or 7.00 s, the host vehicle stops at 1.85 m from the lead car.

7.1.2 | VAE and SVDD training

The data set with the 8160 images used for training the perception LEC is used as the proper training data set. In addition, using simulations with the same random parameters, we collect 2040 images for the calibration set. It should be emphasised that the proper training set and the calibration set should be reshuffled before training the VAE or SVDD. We use a VAE with four layers of 32/64/128/256 × (5 × 5) filters with exponential linear unit (ELU) activations and 2 × 2 max-pooling, one fully connected layer of size 1568 with ELU activation, 1024 latent space, and a symmetric deconvolutional decoder. A simple two-phase learning schedule is employed with initial searching learning rate $\eta = 10^{-4}$ for 250 epochs,

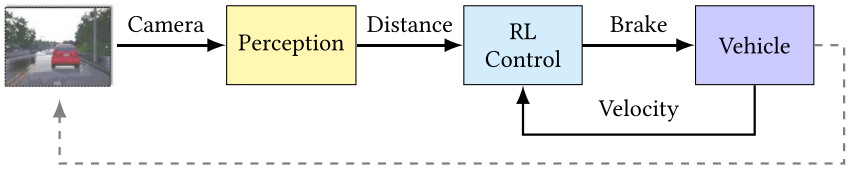


FIGURE 4 Advanced emergency braking system architecture.

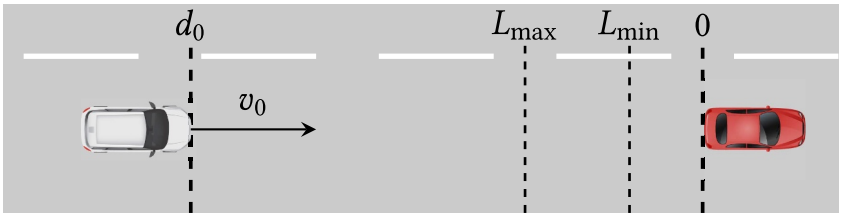


FIGURE 5 Illustration of advanced emergency braking system.

and subsequently fine-tuning $\eta = 10^{-5}$ for 100 epochs. This model is used in the VAE-based out-of-distribution detection method both with and without the saliency maps in order to compute the nonconformity measure.

The deep SVDD is similar with four convolutional layers of 32/64/128/256 × (5 × 5) filters with ELU activations and 2 × 2 max-pooling, followed by one fully connected layer of

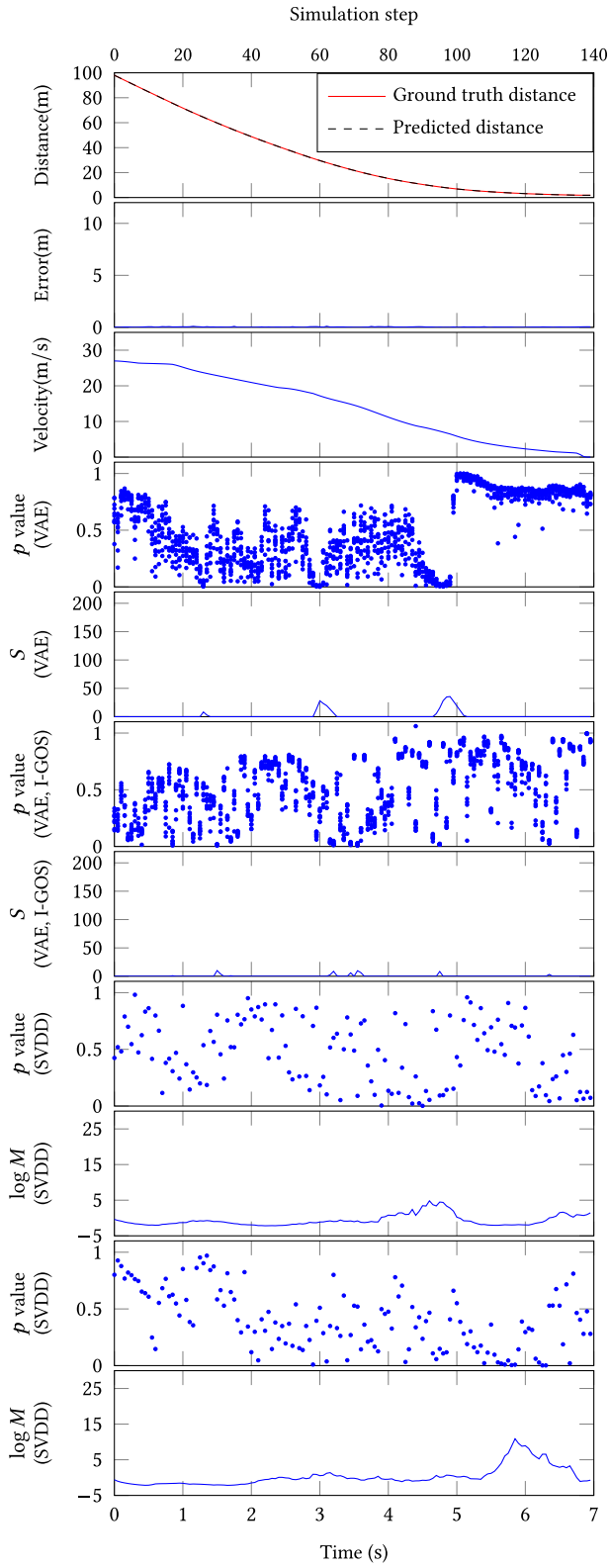


FIGURE 6 Episode with in-distribution inputs.

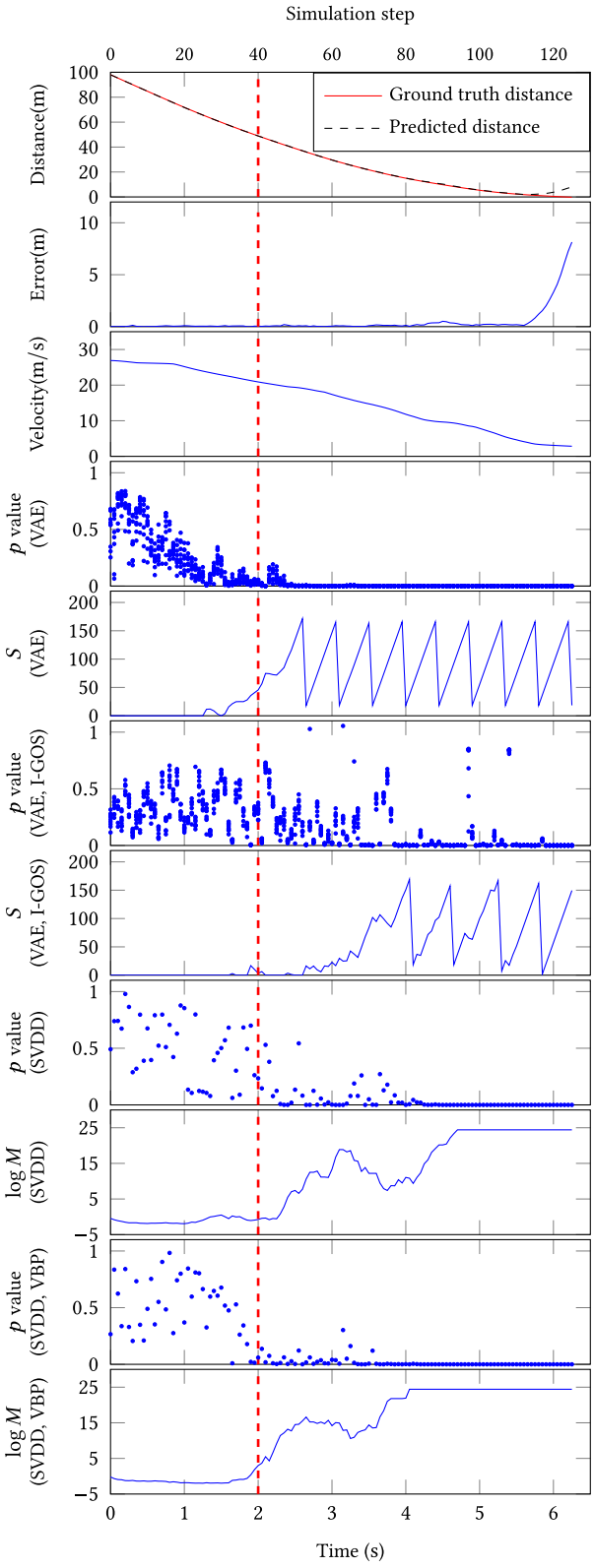


FIGURE 7 Episode with out-of-distribution inputs.

1568 units. As suggested in [29], we first train a deep convolutional autoencoder (DCAE) to initialise the deep SVDD. After 250 ($\eta = 10^{-4}$) + 100 ($\eta = 10^{-5}$) epochs of DCAE training, we copy the weights to the SVDD and set the hypersphere centre $\boldsymbol{c}$ to the mean of the reduced space of the initial forward inference. The one-class deep SVDD objective is used as the loss, and the neural network is trained for additional 150 ($\eta = 10^{-4}$) + 100 ($\eta = 10^{-5}$) epochs. Since the inputs to the SVDD-based detector are different for the case saliency maps are used, two different SVDD networks are trained using the original inputs and saliency-masked inputs respectively.

7.1.3 | Experimental results

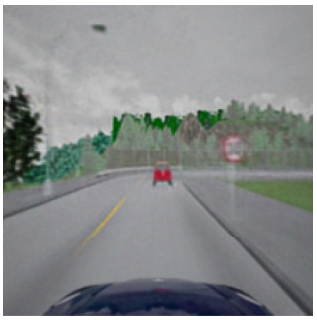
To characterise the performance of the out-of-distribution detection, we use multiple simulation episodes that include in- and out-of-distribution examples. Each episode starts with a random initial velocity v_0 of the host car. The AEBS is activated upon detection of the lead car by the camera as implemented in CARLA. We vary the precipitation parameter r as

$$r = \begin{cases} r_0 & \text{for} \quad t < t_0 \\ r_0 + \beta(t - t_0) & \text{for} \quad t_0 \le t \le t_1 \\ r_0 + \beta(t_1 - t_0) & \text{for} \quad t > t_1 \end{cases}$$

where r_0 is the initial precipitation uniformly sampled from [0, 10]; $t_0 \in \{10, 11, \ldots, 30\}$ is selected randomly as the time step the precipitation starts to increase; $t_1 \in \{90, 91, \ldots, 110\}$ is selected randomly as the time step the precipitation stops increasing; and $\beta \in [0.1, 0.5]$ is a randomly selected slope. In some episodes r is always below 20 (in-distribution), while in other episodes r exceeds 20 and it is assumed that the perception LEC receives out-of-distribution inputs. We simulate 200 episodes, and 108 of them are in-distribution while 92 of them contain out-of-distribution inputs. An in-distribution and an out-of-distribution example are illustrated in Figure 7a,c respectively.

In order to show that the VAE and deep SVDD models are trained successfully and can be used for out-of-distribution detection, we plot the distributions of the nonconformity scores using different nonconformity measures (VAE- and SVDD-based using or not using saliency maps) in Figure 8. More specifically, for VAE-based nonconformity measures, an input is fed into the VAE model to generate a single example, and the nonconformity score is computed as the reconstruction error between the input and generated example (if saliency map is used, the nonconformity score should be weighted by the saliency map). The reconstructed images for both in- and out-of-distribution inputs are shown in Figure 7b,d respectively. From the figures we can see that, since the VAE is trained over the data with no or little rain, it cannot generalise well in the scenario with high rain, and some details in such

(a) Original image (in distribution, precipitation parameter: 10)



(b) Reconstructed image (in distribution, precipitation parameter: 10)

(c) Original image (out of distribution, precipitation parameter: 50)

(d) Reconstructed image (out of distribution, precipitation parameter: 50)

FIGURE 8 Original images and its reconstructed images for both in- and out-of-distribution data in advanced emergency braking system.

scenario, such as the rain on the ground, are not well reconstructed, which then will result in a large nonconformity score. For SVDD-based nonconformity measures, the SVDD model maps the input to the hypersphere, and the nonconformity score is the distance between the representation to the centre (if saliency map is used, the input to SVDD model should be masked by the saliency map).

From the plots, we can see that, for all different nonconformity measures, the nonconformity scores of in-distribution data are much smaller than that of out-of-distribution data. Normally, a threshold is defined on the nonconformity scores to distinguish the in- and out-of-distribution data: if the nonconformity score of an instance is greater than the predefined threshold, such an instance will be identified as an out-of-distribution data, and vice versa. If there is no overlap between the nonconformity scores for in- and out-of-distribution data, it means that all nonconformity scores for in-distribution data are less than those for out-of-distribution data, so there must be a threshold that can completely separate in- and out-of-distribution data, which will result in zero false alarms. The overlapping area of the distributions is inevitable since the precipitation parameter is increased gradually during our data collection, which results in some out-of-distribution data having small changes from the in-distribution data. It means that we cannot find a threshold that can entirely separate in- and out-of-distribution data, and there must be false alarms during the detection. Besides, the greater the overlap rate, the more difficult to distinguish in- and out-of-distribution data, which in turn leads to a higher false alarm rate. Therefore, we measure and report the overlap rate in each plot to indicate the performance of the nonconformity measure. Comparing Figure 8a with Figure 8b, the SVDD-based nonconformity measures have better performance than the VAE-based nonconformity measures. Moreover, in order to compare the performance between the methods using one example and multiple examples, we also plot the distributions of the logarithm of martingales using different nonconformity measures (VAE- and SVDD-based using or not using saliency maps) in Figure 9. In VAE-based methods, for each input, multiple examples are generated, and the martingale is computed based

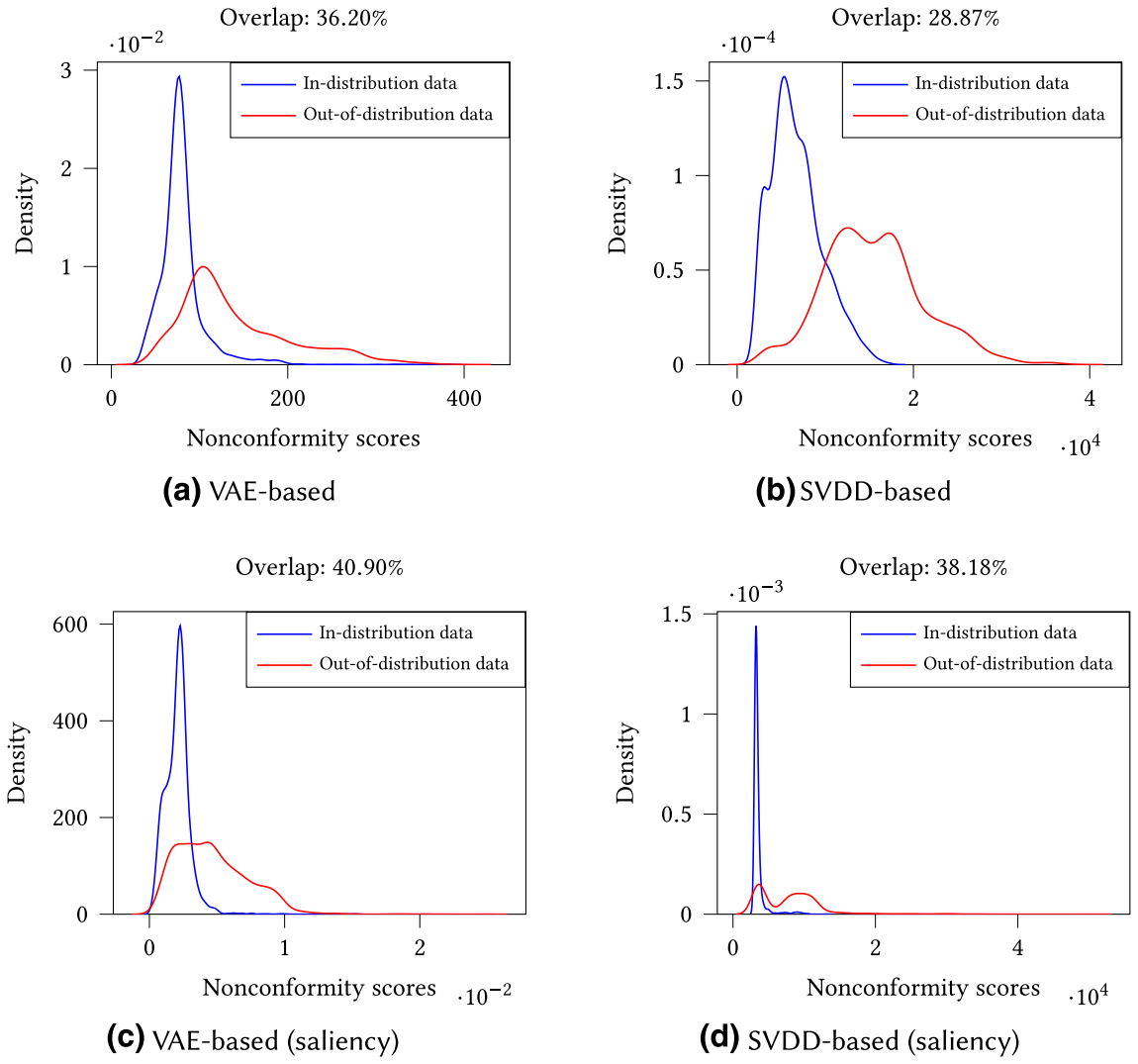


FIGURE 9 Distributions of the nonconformity scores in advanced emergency braking system.

on the corresponding p-values. As for SVDD-based methods, the martingale value is computed by applying the sliding window technique. The overlapping area in Figure 9 using multiple examples are smaller than ones of corresponding plots in Figure 8 using a single example, which demonstrates that the performance of the detector can be improved by incorporating multiple examples.

We illustrate the approach using two episodes, and we plot the ground truth and the predicted distance to the lead car, the velocity of the host car, the p-value of the VAE-based method, and the output of the detector S computed using the logarithm of M_t and $\delta = 6$. Since M_t takes very large values, log M_t is used. We also plot the p-value of the SVDD-based method and the logarithm of the SVDD-based martingale.

The results of the VAE- and SVDD-based methods with saliency maps are plotted similarly. Figure 10 shows one frame of the camera image and its saliency maps generated by I-GOS and VBP. Due to space limitations, we include only the results of the VAE-based using I-GOS and the SVDD-based method using VBP. We use $N = 10$ for the number of examples generated by the VAE. For the SVDD-based method, in order to reasonably choose the size of the sliding window, we perform the auto-correlation analysis: we measure the auto-correlation coefficients on the nonconformity scores of the input sequence and compute the mean absolute auto-correlation within different time scales, which is shown in Figure 11. The auto-correlation becomes smaller than 0.1 when the time scale is greater than 9. Therefore, we can choose the time scale greater than 9. In the illustrative episode, the size of the sliding window is set to 10.

Figure 6 shows simulation results for the in-distribution case. Most of the p-values are much greater than 0, and the martingale for all four approaches is small. The VAE-based method is more sensitive than the SVDD as indicated by the larger value around 5 s. In this scenario, there is a speed limit traffic sign that is not accurately reconstructed by the VAE resulting in smaller p-values. After the car passes the traffic sign, the p-values increase, and the martingale decreases. Such effect can be attenuated by using SVDD-based method or

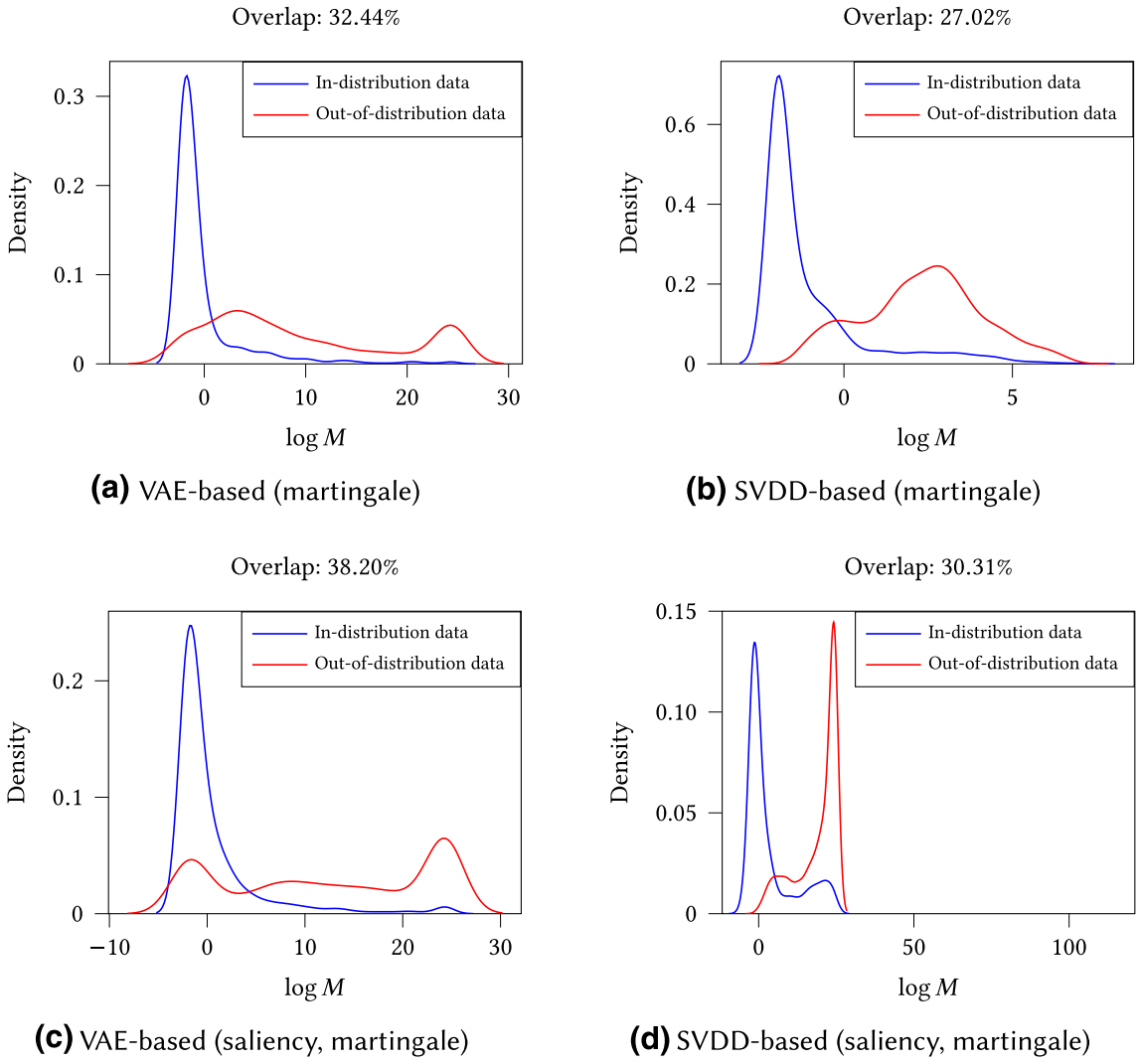


FIGURE 10 Distributions of the martingales in advanced emergency braking system.

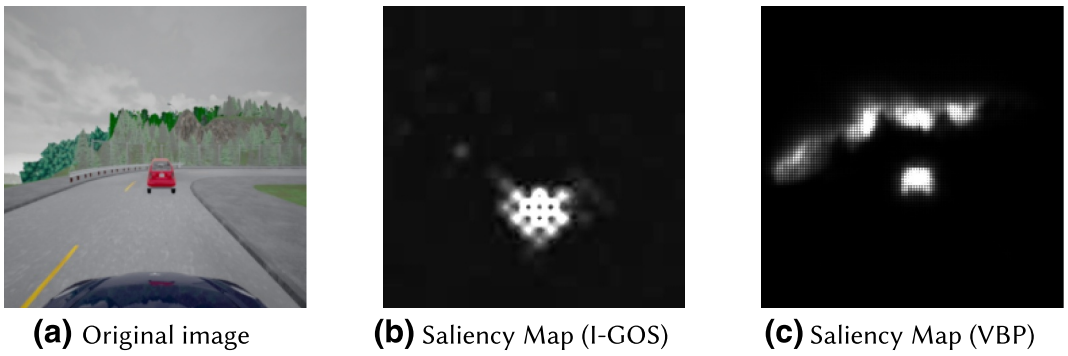


FIGURE 11 Original image and its saliency maps for advanced emergency braking system.

introducing saliency map to the VAE-based method. The reasons are the SVDD-based method does not use a reconstructed image for nonconformity computing, while the saliency map focuses the computation on the features that contribute most to the output of the LEC.

An episode with out-of-distribution inputs is shown in Figure 12. The parameter r exceeds 20 at time step 40 (2.0 s). The error of the perception LEC starts increasing and reaches almost 9 m. The controller is misled by the perception LEC and does not stop the car which collides with the lead car (velocity is greater than 0 when ground truth distance comes to 0). For all four detection methods, the martingale grows as the p-values become smaller.

We evaluate the approach for the 200 episodes generated by considering different values of N. We run each episode, and if an alarm is raised, we stop the simulation, and we check if the alarm is false. We compute the detection delay as the number of frames from the time r exceeds 20. We select the detector parameters τ and δ using a simple search for achieving the average detection delay less than 25 frames and the number of false alarms less than 2. It should be emphasised that we can choose the detector parameters only based on the nonconformity scores of in-distribution data as the other unsupervised detection method. However, in our method, some out-of-distribution data can be collected to improve the detector's performance. With more out-of-distribution episodes collected, the parameters can be better tuned, and the detector's performance can be further improved. Table 1 reports the results for the VAE- and SVDD-based methods. Furthermore, in order to compare the performance of the detectors, we use the same parameters for the methods with saliency maps, and Table 1 summarises the results. The number of false alarms is very small, and the delay for detection is smaller than 20 frames or 1 s for all four detection methods. The VAE-based method uses a cumulative sum (CUSUM) procedure, which is parameterised by δ and τ. These two parameters control the trade-off among false positive, false negative, and detection delay. The large value of σ and τ will result in the detector under-sensitive, and the detector will have a large number of false negatives and long detection delay. On the other side, the small value of σ and τ will have an over-sensitive detector. Similar to the VAE-based method, the SVDD-based method is

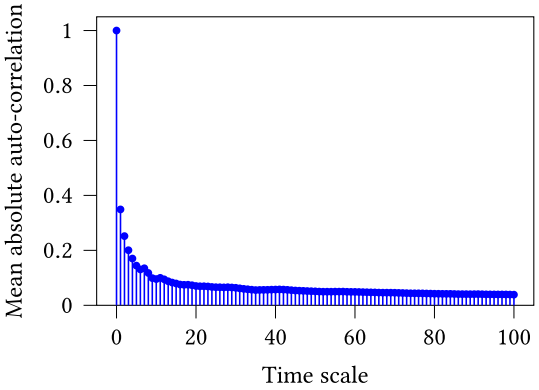


FIGURE 12 Mean absolute auto-correlation of the nonconformity scores in advanced emergency braking system.

parameterised by the size of sliding window N and the threshold of stateless detector τ. The large value of N and the small value of τ will lead to an under-sensitive detector, and vice versa. The overlap rate in Figure 9a is around 32%, which indicates that the false alarm rate is quite large when we directly use the martingale value for detection without a CUSUM detector. The false alarms of VAE-based method reported in Table 1 drops to almost zero by introducing the CUSUM detector, demonstrating that the CUSUM detector can improve the performance of the detector.

From the results for the methods with saliency maps, it can be seen that the detector is more robust to small variations in the input that do not affect the LEC prediction (and the prediction error). For example, when the precipitation parameter exceeds slightly the defined threshold for out-of-distribution inputs (20), the prediction error remains very small. Figure 12 shows that for small changes from the in-distribution data, the p-values are larger in the case of saliency maps, and therefore, out-of-distribution detection occurs later which is more consistent with the prediction error. The results are similar for the other experiments as summarised in the increased average delay in Tables 1–4. The detection delay in the methods with saliency maps is slightly greater, which is because the detection methods with saliency maps consider the influence of the input on the LEC output.

TABLE 1 False alarms and average delay for advanced emergency braking system

Methods	Parameters $(N, \sigma, \tau)/(N, \tau)$	False positive	False negative	Average delay (frames)
VAE	5, 5, 42	2/108	0/92	17.91
	5, 5, 49	0/108	0/92	19.84
	10, 6, 156	0/108	0/92	18.65
	10, 10, 106	0/108	0/92	19.30
	20, 16, 250	2/108	0/92	17.63
	20, 18, 240	0/108	0/92	18.46
VAE + IGOS	5, 5, 42	1/108	0/92	22.32
	5, 5, 49	1/108	0/92	24.04
	10, 6, 156	0/108	0/92	23.36
	10, 10, 106	0/108	0/92	23.70
	20, 16, 250	1/108	0/92	21.14
	20, 18, 240	0/108	0/92	22.83
SVDD	10, 12	1/108	0/92	14.38
	10, 14	0/108	0/92	17.78
	15, 13	2/108	0/92	13.48
	15, 15	0/108	0/92	15.36
	20, 16	1/108	0/92	12.02
	20, 17	0/108	0/92	13.29
SVDD + VBP	10, 12	2/108	0/92	15.32
	10, 14	1/108	0/92	18.10
	15, 13	1/108	0/92	14.68
	15, 15	0/108	0/92	16.42
	20, 16	1/108	0/92	13.28
	20, 17	0/108	0/92	14.32

TABLE 2 Missed alarms and average delay for advanced emergency braking system

Methods	Parameters $(N, \sigma, \tau)/(N, \tau)$	Missed alarms (False negative)	Average delay (frames)
VAE	5, 6, 43	1/92	19.54
	10, 8, 132	0/92	19.46
	20, 19, 235	1/92	19.02
VAE + I-GOS	5, 5, 53	1/92	24.08
	10, 8, 130	2/92	24.05
	20, 20, 232	0/92	23.96
SVDD	10, 15	1/92	18.12
	15, 14	0/92	14.70
	20, 18	0/92	13.94
SVDD + VBP	10, 15	1/92	19.21
	15, 14	0/92	15.78
	20, 18	1/92	14.76

TABLE 3 False alarms and average delay for real-world perception neural network

Methods	Parameters $(N, \sigma, \tau)/(N, \tau)$	False positive	False negative	Average delay (frames)
VAE	5, 3, 36	0/50	0/50	11.32
	5, 5, 15	0/50	1/50	13.63
	10, 5, 105	0/50	0/50	12.14
	10, 8, 55	0/50	0/50	10.78
	20, 10, 235	0/50	0/50	12.18
	20, 17, 140	0/50	0/50	11.30
VAE + I-GOS	5, 3, 36	0/50	0/50	13.42
	5, 5, 15	0/50	0/50	16.10
	10, 5, 105	0/50	0/50	15.08
	10, 8, 55	0/50	0/50	12.74
	20, 10, 235	0/50	0/50	15.32
	20, 17, 140	0/50	0/50	14.12
SVDD	10, 4	0/50	0/50	10.62
	10, 5	0/50	0/50	12.58
	15, 4	0/50	0/50	11.24
	15, 6	0/50	0/50	12.68
	20, 5	0/50	0/50	10.14
	20, 6	0/50	0/50	11.66
SVDD + sVBP	10, 4	0/50	0/50	11.46
	10, 5	0/50	0/50	13.32
	15, 4	0/50	0/50	11.34
	15, 6	0/50	0/50	12.78
	20, 5	0/50	0/50	10.74
	20, 6	0/50	0/50	12.32

For example, when the precipitation parameter just exceeds 20 the influence to the LEC output (and the prediction error are very small) and additional frames are required for detecting out-of-distribution inputs. In addition, missed alarms is an attractive performance metric for such a detection system in CPS. Therefore we tune the parameters achieving 0 false positives and report the missed alarms (false negatives) in Table 2. The results reveal that our approach has a small number of missed alarms for different values of N.

7.2 | Self-driving end-to-end control

7.2.1 | Experimental setup

The CARLA simulator comes with a self-driving end-to-end controller trained using imitation learning. The SDEC uses camera images as inputs and computes steering, acceleration,

and brake actuation signals applied to the car. The architecture is shown in Figure 13.

The SDEC is implemented using a CNN trained by conditional imitation learning with 14 h of driving data recorded by human drivers [10]. The sampling period used here is $\Delta t = 1/10$ s. For this example, our objective is to evaluate if the method can be used to detect a class of adversarial attacks. An approach for designing physically realisable attacks in end-to-end autonomous driving is presented in [8], and a novel class of hijacking attacks is introduced where painted lines on the road cause the self-driving car to follow a target path. Figure 14b shows an image with the painted pattern on the road.

In order to train the VAE and SVDD, we collect training data using episodes without attacks. First, we generate 633 images in two different weather patterns (clear noon and cloudy noon) and three different scenarios (turning right, turning left, and going straight). Then, we reshuffle and randomly split the training data into 506 images for the proper training data set and 127 images for the calibration set. We use

TABLE 4 Missed alarms and average delay for real-world perception neural network

Methods	Parameters $(N, \sigma, \tau)/(N, \tau)$	Missed alarms (False negative)	Average delay (frames)
VAE	5, 4, 24	1/50	12.48
	10, 6, 90	0/50	11.68
	20, 15, 180	0/50	12.02
VAE + I-GOS	5, 4, 24	0/50	15.04
	10, 6, 92	0/50	14.38
	20, 15, 175	0/50	14.68
SVDD	10, 6	1/50	13.21
	15, 5	0/50	11.74
	20, 7	0/50	12.10
SVDD + VBP	10, 6	0/50	13.75
	15, 5	0/50	12.22
	20, 7	0/50	12.84

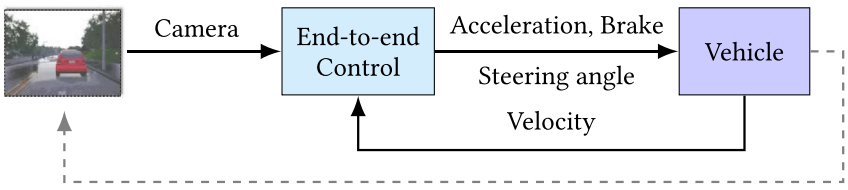


FIGURE 13 Self-driving end-to-end control system architecture.

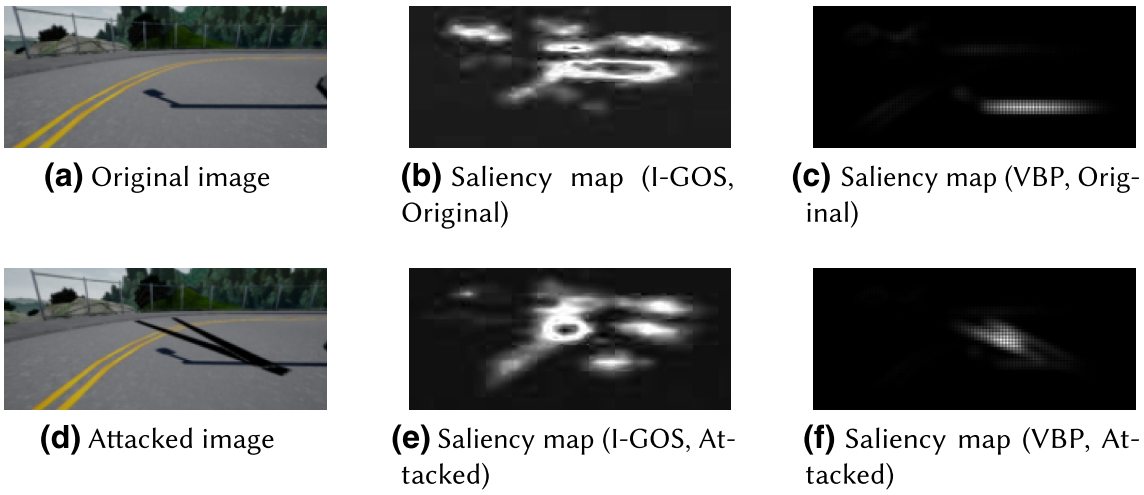


FIGURE 14 Original image, physical adversarial image and their saliency maps for self-driving end-to-end control.

the same VAE and SVDD architectures and hyperparameters as in the AEBS.

7.2.2 | Experimental results

The evaluation focuses on the *Right Corner Driving* case, which is reported as more vulnerable [8]. We run 105 simulation episodes described in ref. [8] with different attacks such as positions and rotations of the two black lines which are chosen to cause traffic infractions. In 69 out of the 105 episodes, the attack is successfully causing a vehicle crash. Our approach detects the attacks in all 105 episodes. We plot the p-values and detector output S of the VAE-based method, the p-values and the logarithm of the SVDD-based martingale for episodes with and without attacks shown in Figure 15 and 16 respectively. The size of the sliding window N for SVDD-based method is chosen based on the auto-correlation analysis on the nonconformity scores of the input sequence. The mean absolute auto-correlation is smaller than 0.1 when the time scale is greater than 7, and therefore, we can choose $N = 10$ in our illustrative episodes. In addition, Figures 15 and 16 show the results of the detection methods using saliency maps. Figure 14 shows the original (in-distribution) image, physical adversarial image, and their saliency maps. For the in-distribution (no-attack) episode, most of the p-values are much greater than 0, while the martingales for all approaches are small. In the adversarial episode, there are two black lines painted on the road as shown in Figure 14d, and the vehicle is misled to a crash. The p-values are almost 0, and the martingales grow very large, indicating the input images are out-of-distribution.

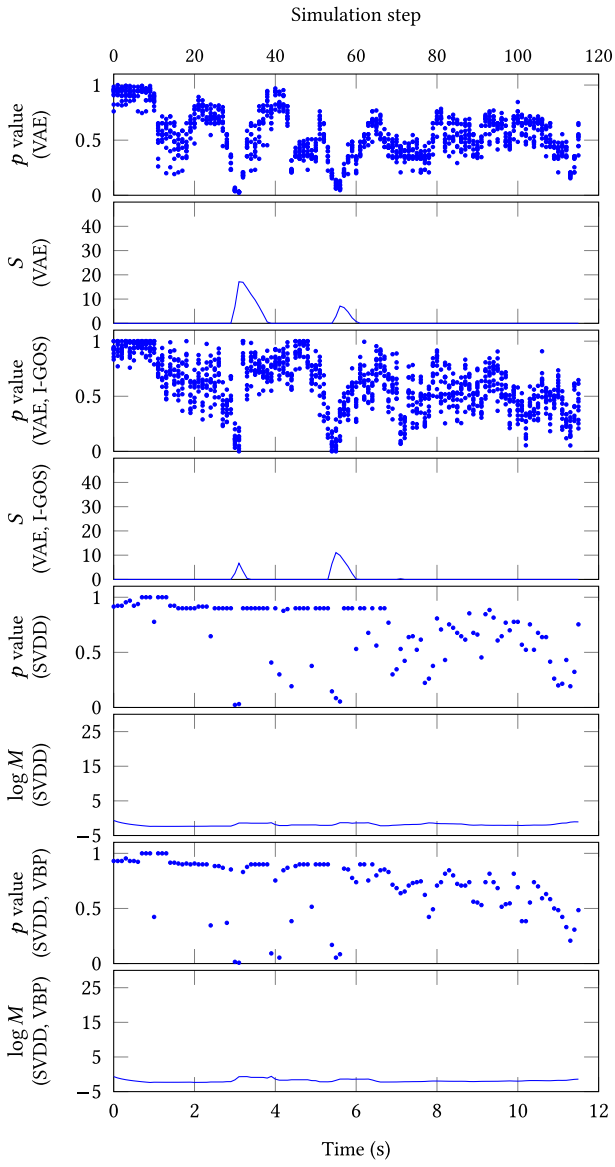


FIGURE 15 Episode with in-distribution inputs.

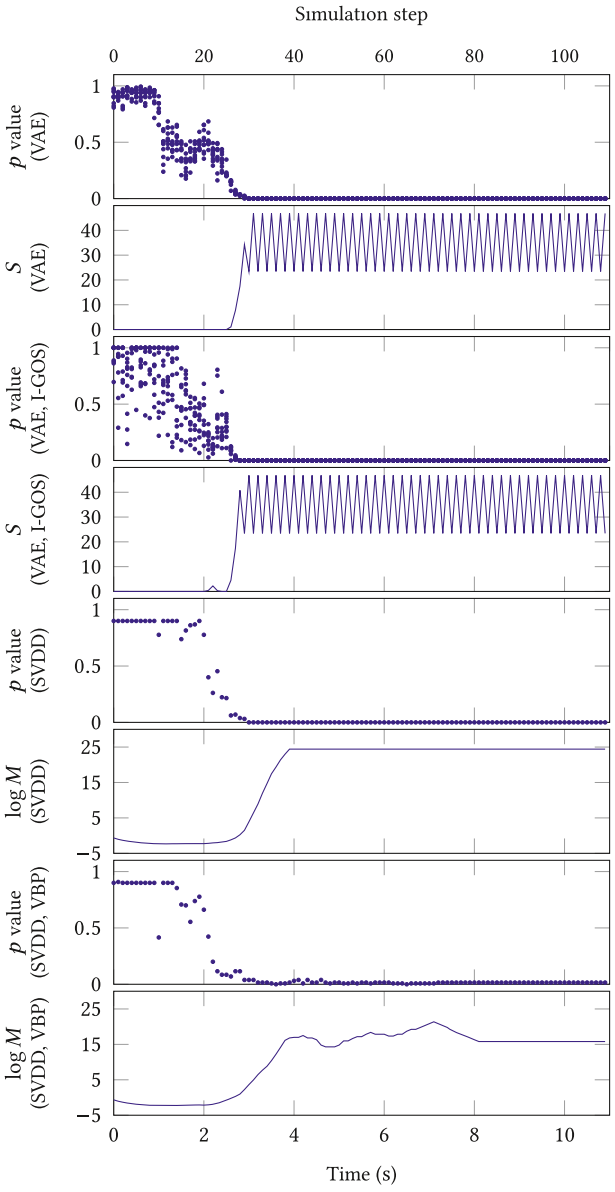


FIGURE 16 Episode with attacked inputs.

7.3 | Autonomous vehicle seasonal dataset

7.3.1 | Experimental setup

In order to evaluate the approach in a real-world environment, we use the Ford autonomous vehicle seasonal data set [1], which is used to train a perception neural network whose task is to predict the heading changes of the host vehicle from the images captured by a camera. The sampling rate of the camera is $\Delta t = 1/7$ s. The data set provides the raw images and ground truth pose (position and orientation) of the vehicle in a global frame. The images are collected under seasonal variation in weather and include various lighting, construction, and traffic conditions experienced in typical urban environments [1]. For using the data set to evaluate our approach, we pre-process the data to compute heading changes of the host vehicle by converting quaternions to Euler angles and calculating the yaw difference, and we synchronise the input images with the heading changes.

We select the data set for one vehicle (V1) and one drive (Log1) as the training data set. The set contains data collected in cloudy weather and freeway, overpass, and bridge drive scenarios. We reshuffle and randomly split the training data into 3556 images for the proper training data set and 889 for the calibration set. The proper training data set is used to train the perception, VAE, and SVDD networks.

The perception network is a CNN whose architecture is similar to the NVIDIA end-to-end self-driving controller [7]. After 100-epoch training, the mean absolute error for training and testing are 0.012° and 0.016° respectively. The VAE and SVDD networks used for detection use the same architectures and hyperparameters as in the AEBS and SDEC examples.

7.3.2 | Experimental results

We evaluate the approach using 100 episodes, 50 of which are in-distribution and 50 out-of-distribution. For out-of-distribution data, we use a set (Log3) with data collected in sunny weather and residential driving scenario. Each episode contains 140 sequential frames and has duration 20 s. We illustrate the approach using two episodes, and we plot the prediction errors of the heading change, the p-values and detector output of the VAE-based method with and without saliency maps, and the p-values and the logarithm of the SVDD-based martingale with and without saliency maps. The results are shown in Figure 17 and Figure 18 respectively. Similar to the AEBS and SDEC, in order to choose a reasonable size of sliding window N for SVDD-based method, we analyse the auto-correlation function on the nonconformity scores of the input sequence. The mean absolute auto-correlation is smaller than 0.1 when the time scale is greater than 6. In our illustrative episode, we choose $N = 10$.

We also show the in- and out-of-distribution input images and their saliency maps in Figure 19. For the in-distribution episode, most of the p-values are far away from 0, and the martingales in all approaches are small indicating there is no out-of-distribution input detected. In the out-of-distribution episode, the predicted errors are greater than the in-distribution episode. For all approaches, the p-values are small, and the martingales grow very large showing the input images are not in the same distribution as the training data set.

We also report the number of false alarms and average detection delay time in Table 3, and the number of missed alarms in Table 4 by considering different values of N and detector parameters σ and τ. From the results, the number of false alarms and missed alarms are very small and the delay for

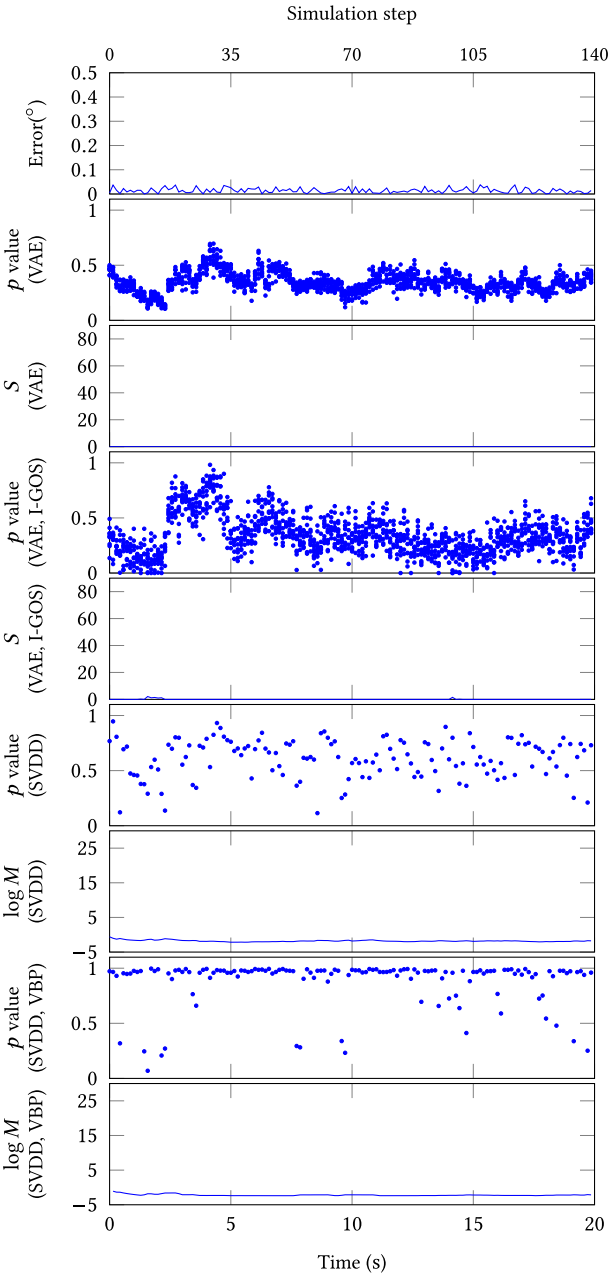


FIGURE 17 Episode with in-distribution inputs.

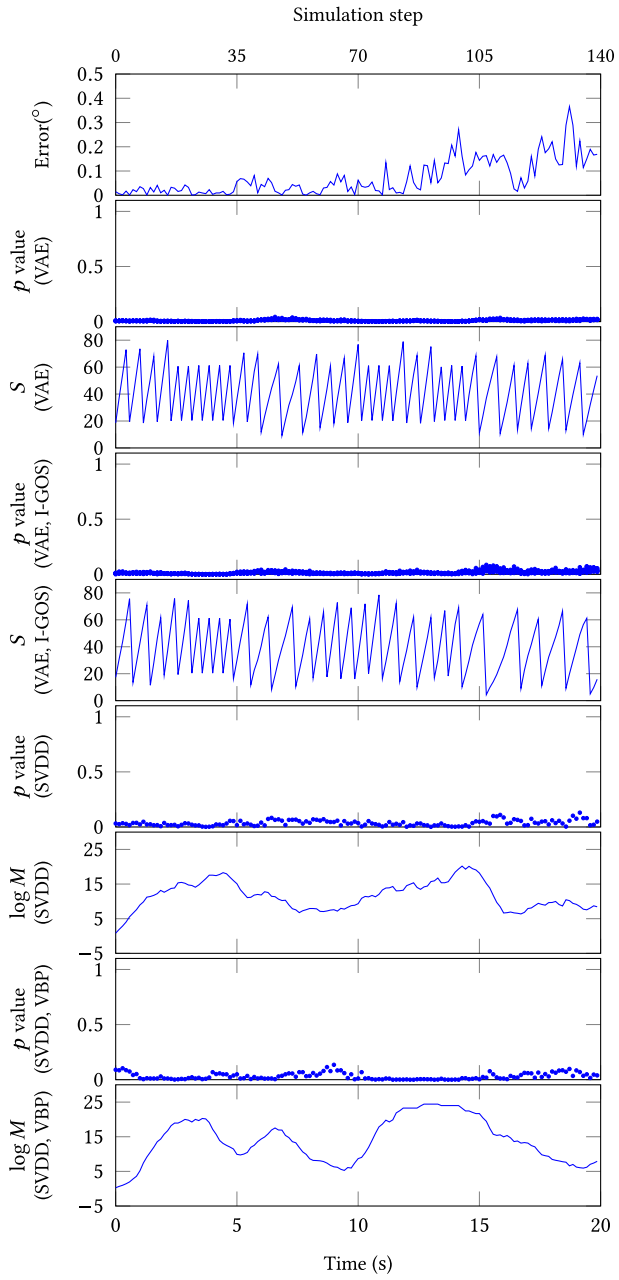


FIGURE 18 Episode with out-of-distribution inputs.

detection is smaller than 20 frames. The results also show that the SVDD-based methods have short detection delay than the VAE-based methods.

7.4 | Computational efficiency

The VAE-based and SVDD-based methods can compute the nonconformity scores in real-time without storing training data. Table 5 reports the minimum (min), first quartile (Q_1), second quartile or median (Q_2), third quartile (Q_3), and maximum (max) of (1) the execution times of the LECs in AEBS, SDEC, and AVSD, (2) the execution times of the I-GOS- and VBP-based saliency map algorithms, and (3) the execution times of the VAE-based and SVDD-based detectors with and without saliency maps for different values of N.

The results show that the VBP-based algorithm is slightly more efficient than the I-GOS. Since the VAE-based method uses N examples in each time step to compute the nonconformity scores, the execution time is larger than the execution time of the SVDD-based method. The execution time of SVDD-based detection method is independent of the window size N since the martingale can be computed recursively for the sliding window. The execution times are similar to the execution times of the perception and end-to-end control LECs and much smaller than the corresponding sampling time (50 ms in AEBS, 100 ms in SDEC and 1000/7 ms in AVSD), and thus, the methods can be used for real-time out-of-distribution detection.

Furthermore, in order to demonstrate current techniques struggle to cope with high-dimensional inputs, we apply the k-nearest neighbour (k-NN) nonconformity measure [26] for detecting out-of-distribution data in the experiment of advanced emergency braking system. The k-NN nonconformity measure is defined as the sum of the distances between the test example and its k nearest neighbours in the proper training dataset,

$$\alpha_{k\text{-NN}} = \sum_{i=1}^{k} \text{dist}_i,$$

where dist_i is the distance from test example to ith nearest examples in the proper training dataset, and we use the Euclidean distance as the distance metric between two images. Given 8160 images with a resolution 224 × 224 × 3 in the proper training dataset and $k = 10$, the execution time is around 2 s, which is much more greater than the sampling time of braking system, and therefore, such nonconformity measure cannot scale to high-dimensional inputs.

8 | CONCLUSION

In this study, we demonstrated a method for out-of-distribution detection in learning-enabled CPS. The method is based on inductive conformal prediction and anomaly detection but uses VAEs and SVDD to learn models to efficiently compute the nonconformity of new inputs relative to the training set and enable real-time detection of high-dimensional out-of-distribution inputs. In addition, the saliency maps can be incorporated to improve the robustness of the detector. Our evaluation is based on two simulation case studies of an AEBS and an SDEC as well as a real-world data set for autonomous driving. The results demonstrate a very small number of false positives and detection delay while the execution time is comparable to the execution time of the original LECs. Promising future work is to combine the outputs of the neural network into out-of-distribution detection. Another possible direction is to explore the physically realisable adaptive adversarial attacks and to adapt our approach to such attacks.

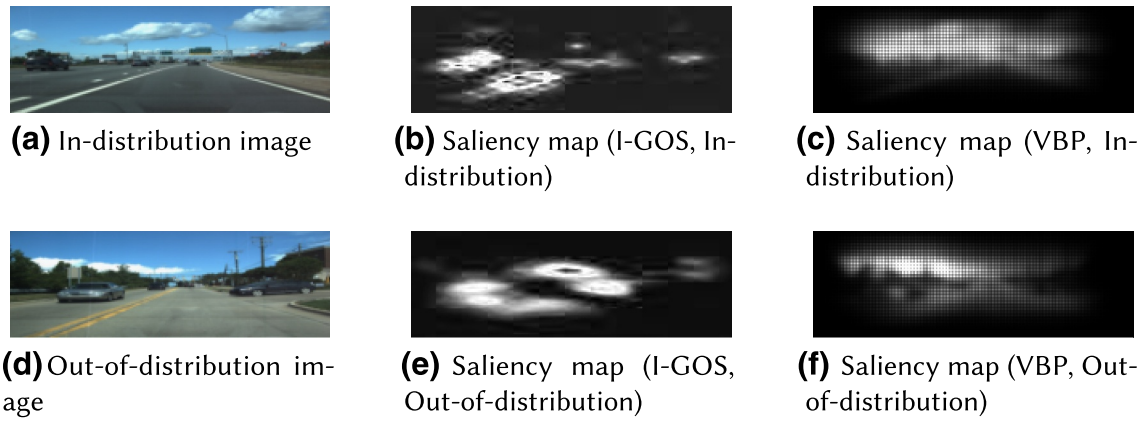


FIGURE 19 In- and out-of-distribution images and their saliency maps for real-world perception neural network.

TABLE 5 Execution times

	N	min (ms)	Q_1 (ms)	Q_2 (ms)	Q_3 (ms)	max (ms)
AEBS	N/A	3.48	3.85	3.91	3.96	4.20
SDEC	N/A	2.20	2.37	2.45	2.36	3.31
RPNN	N/A	0.51	0.52	0.52	0.53	0.56
k-NN	N/A	1843	2093	2098	2113	2287
I-GOS	N/A	4.32	4.37	4.41	4.43	4.47
VBP	N/A	1.37	1.38	1.38	1.39	1.41
VAE	5	15.43	15.47	15.49	15.50	15.60
	10	31.33	31.41	31.43	31.45	31.77
	20	64.54	64.70	64.72	64.75	65.00
VAE + I-GOS	5	20.18	20.21	20.24	20.29	20.32
	10	36.44	36.46	36.49	34.52	34.61
	20	70.54	70.59	70.64	70.71	70.84
SVDD	10	2.12	2.35	2.44	2.45	2.52
	15	2.16	2.23	2.34	2.41	2.49
	20	2.28	2.33	2.34	2.59	2.60
SVDD + VBP	10	3.43	3.49	3.50	3.65	3.70
	15	3.46	3.51	3.60	3.68	3.79
	20	3.53	3.57	3.69	3.70	3.82

ACKNOWLEDGEMENTS

The material presented in this study is based upon work supported by the National Science Foundation (NSF) under grant number CNS 1739328, and the Defense Advanced Research Projects Agency (DARPA) through contract number FA8750-18-C-0089. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, or DARPA.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interests.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Feiyang Cai https://orcid.org/0000-0002-1486-0971

REFERENCES

1. Agarwal, S., et al.: Ford Multi-AV Seasonal Dataset (2020). arXiv preprint (2020)
2. An, J., Cho, S.: Variational Autoencoder Based Anomaly Detection Using Reconstruction Probability. Special Lecture on IE (2015)

3. Bach, S., et al.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS One 10, 7, e0130140 (2015). https://doi.org/10.1371/journal.pone.0130140
4. Balasubramanian, V., Ho, S.-S., Vovk, V.: Conformal Prediction for Reliable Machine Learning: Theory, Adaptations and Applications. Morgan Kaufmann Publishers Inc (2014)
5. Basseville, M., Nikiforov, I.V.: Detection of Abrupt Changes: Theory and Application. Prentice-Hall, Inc (1993)
6. Bojarski, M., et al.: VisualBackProp: efficient visualization of CNNs for autonomous driving. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '18) (2018)
7. Bojarski, M., et al.: End to End Learning for Self-Driving Cars (2016). arXiv preprint (2016)
8. Boloor, A., et al.: Attacking vision-based perception in end-to-end autonomous driving models. J. Syst. Architect. 110, 101766 (2020). https://doi.org/10.1016/j.sysarc.2020.101766
9. Carlini, N., Wagner, D.A.: Towards evaluating the robustness of neural networks. In: IEEE Symposium on Security and Privacy, pp. 39–57. IEEE Computer Society (2017)
10. Alexey Dosovitskiy, et al.: CARLA: an open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning (CoRL '17) (2017)
11. Dreossi, T., Alexandre, D., SanjitSeshia, A.: Compositional falsification of cyber-physical systems with machine learning components. J. Autom. Reas. 63(4), 1031–1053 (2019). https://doi.org/10.1007/s10817-018-09509-5
12. Eykholt, K., et al.: Robust physical-world attacks on deep learning visual classification. In: Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '2018) (2018)
13. Fedorova, V., et al.: Plug-in martingales for testing exchangeability online. In: Proceedings of the 29th International Conference on Machine Learning (ICML '12) (2012)
14. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR '15) (2015)
15. Gu, X., Easwaran, A.: Towards safe machine learning for CPS: infer uncertainty from training data. In: Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS '19) (2019)
16. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: Proceedings of the 5th International Conference on Learning Representations (ICLR '17) (2017)
17. Hendrycks, D., et al.: Deep anomaly detection with outlier exposure. In: Proceedings of the 7th International Conference on Learning Representations (ICLR '19) (2019)
18. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: Proceedings of the 2nd International Conference on Learning Representations (ICLR '14) (2014)
19. Laxhammar, R., Falkman, G.: Online learning and sequential anomaly detection in trajectories. IEEE Trans. Pattern Anal. Mach. Intell. 36(6), 1158–1173 (2014). https://doi.org/10.1109/tpami.2013.172
20. Laxhammar, R., Falkman, G.: Inductive conformal anomaly detection for sequential detection of anomalous sub-trajectories. Ann. Math. Artif. Intell. 74(1-2), 67–94 (2015). https://doi.org/10.1007/s10472-013-9381-7
21. Liang, S., Li, Y., Srikant, R.: Enhancing the reliability of out-of-distribution image detection in neural networks. In: Proceedings of the 6th International Conference on Learning Representations (ICLR '18) (2018)
22. Lillicrap, T.P., et al.: Continuous control with deep reinforcement learning. In: Proceedings of the 4th International Conference on Learning Representations (ICLR '16) (2016)
23. McAllister, R., et al.: Robustness to out-of-distribution inputs via task-aware generative uncertainty. In: Proceedings of the International Conference on Robotics and Automation (ICRA '19) (2019)
24. Meng, D., Chen, H.: MagNet: a two-pronged defense against adversarial examples. In: Thuraisingham, B.M., et al. (eds.) Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '2017) (2017)
25. Nguyen, A.M., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: high confidence predictions for unrecognizable images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15) (2015)
26. Papadopoulos, H., Vovk, V., Gammerman, A.: Regression conformal prediction with nearest neighbours. J. Artif. Intell. Res. 40(2011), 815–840 (2011). https://doi.org/10.1613/jair.3198
27. Qi, Z., Khorram, S., Li, F.: Visualizing deep networks by optimizing with integrated gradients. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence (2020)
28. Richter, C., Roy, N.: Safe visual navigation via deep learning and novelty detection. In: Proceedings of Robotics: Science and Systems (RSS '17) (2017)
29. Ruff, L., et al.: Deep one-class classification. In: Proceedings of the 35th International Conference on Machine Learning (ICML '18) (2018)
30. Seshia, S.A., Sadigh, D., Shankar Sastry, S.: Towards Verified Artificial Intelligence (2016). arXiv preprint (2016)
31. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. In: Workshop Track Proceedings of the 2nd International Conference on Learning Representations (2014)
32. Smith, J., et al.: Anomaly detection of trajectories with kernel density estimation by conformal prediction. In: Proceedings of the International Conference on Artificial Intelligence Applications and Innovations (AIAI '14) (2014)
33. Cumhur Erkan, T., Fainekos, G., Ito, H., James, K.: Sim-ATAV: simulation-based adversarial testing framework for autonomous vehicles. In: Proceedings of the 21st International Conference on Hybrid Systems (HSCC '18) (2018)
34. Volkhonskiy, D., et al.: Martingales for change-point detection. In: Proceedings of the Workshop on Conformal and Probabilistic Prediction and Applications (COPA '17) (2017)
35. Vovk, V., Gammerman, A., Glenn, S.: Algorithmic Learning in a Random World. Springer-Verlag (2005)
36. Wang, Z., et al.: Dueling network architectures for deep reinforcement learning. In: Proceedings of the 33nd International Conference on Machine Learning (ICML '16) (2016)
37. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European Conference on Computer Vision, pp. 818–833. Springer (2014)

How to cite this article: Cai, F., Koutsoukos, X.: Real-time out-of-distribution detection in cyber-physical systems with learning-enabled components. IET Cyber-Phys. Syst., Theory Appl. 7(4), 212–234 (2022). https://doi.org/10.1049/cps2.12034