Detecting Adversarial Examples in Learning-Enabled Cyber-Physical Systems using Variational Autoencoder for Regression

Feiyang Cai
Vanderbilt University
Nashville, TN
feiyang.cai@vanderbilt.edu

Jiani Li
Vanderbilt University
Nashville, TN
jiani.li@vanderbilt.edu

Xenofon Koutsoukos
Vanderbilt University
Nashville, TN
xenofon.koutsoukos@vanderbilt.edu

***Abstract*—Learning-enabled components (LECs) are widely used in cyber-physical systems (CPS) since they can handle the uncertainty and variability of the environment and increase the level of autonomy. However, it has been shown that LECs such as deep neural networks (DNN) are not robust and adversarial examples can cause the model to make a false prediction. The paper considers the problem of efficiently detecting adversarial examples in LECs used for regression in CPS. The proposed approach is based on inductive conformal prediction and uses a regression model based on variational autoencoder. The architecture allows to take into consideration both the input and the neural network prediction for detecting adversarial, and more generally, out-of-distribution examples. We demonstrate the method using an advanced emergency braking system implemented in an open source simulator for self-driving cars where a DNN is used to estimate the distance to an obstacle. The simulation results show that the method can effectively detect adversarial examples with a short detection delay.**

***Keywords*-adversarial example detection, inductive conformal prediction, VAE based regression, self-driving vehicles**

I. Introduction

The use of learning-enabled components (LECs) such as deep neural networks (DNNs) is on the rise in many classes of cyber-physical systems (CPS). Semi-autonomous and autonomous vehicles, in particular, are applications where LECs play a significant role. Although DNNs can increase the level of the autonomy by handling the uncertainty and variability of the environment, multiple studies have shown that DNNs are not robust. For example, they are vulnerable to examples with small specially human-crafted perturbations in the input that cause false predictions [1], [2].

LECs rely on design-time data-driven training to learn how to operate in unstructured and dynamic environments. However, LECs must be deployed and operate in a real system and the input data may be different than the data used for training and testing. Adversarial examples, in particular, can cause wrong predictions and impact system safety [3]. Adversarial example detection for CPS must have a low false alarm rate while being computational efficient for real-time monitoring.

Recently, there have been many efforts to defend against adversarial examples especially in the context of classification. Autoencoder based methods, such as [4], [5], utilize the reconstruction error or the reconstruction accuracy to detect adversarial examples. The autoencoders learn and encode the properties of the training data with latent representations. If a test example is from the same distribution as the training set, we expect a small reconstruction error. The work in [6] presents an approach which aims to map the in-distribution input data into a hypersphere of minimum volume characterized by center c and radius R. The adversarial examples are expected to be mapped to points out of the hypersphere. Typically, existing methods do not take into consideration the dynamical behavior.

In our previous work [7], we developed an approach which leverages inductive conformal prediction [8], [9] and inductive anomaly detection [10] for out-of-distribution detection in learning-enabled CPS. The detection algorithm is based on a variational autoencoder (VAE) which is designed independently of the perception LEC, and the approach considers only the input data. Taking into account the predicted output can be beneficial for detection of adversarial examples especially since they are crafted so that a small perturbation of the inputs causes a large change in the output. Instead of training the regression and VAE model used for detection independently, this paper uses a VAE-based regression model which integrates the regression model into the VAE and takes into account the predicted output for adversarial detection.

The main contribution of the paper is an approach for detection of adversarial examples for regression LECs in CPS based on inductive conformal anomaly detection. In order to take into consideration the output of the regression LEC, the proposed approach employs a VAE-based regression model which is learned jointly combining the VAE and regressor

The material presented in this paper is based upon work supported by the National Science Foundation (NSF) under grant numbers CNS 1739328, the Defense Advanced Research Projects Agency (DARPA) through contract number FA8750-18-C-0089, and the Air Force Office of Scientific Research (AFOSR) DDDAS through contract number FA9550-18-1-0126. 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 AFOSR, DARPA, or NSF.

DOI 10.1109/SPW50608.2020.00050

by conditioning the latent representation of the VAE on the target variable of the regressor. The inductive conformal anomaly detection is based on the VAE-based regression model and can be performed efficiently for high-dimensional inputs.

Another contribution is the empirical evaluation of the approach using an advanced emergency braking system (AEBS) implemented in CARLA [11], an open source simulator for self-driving cars. The AEBS uses a perception LEC to detect the nearest front obstacle and estimate the distance from the host vehicle based on the images captured by an onboard camera. The distance estimated by the LEC is used by a reinforcement learning controller together with the vehicle's velocity to compute the desired braking in order to safely stop the vehicle. The fast gradient sign method (FGSM) [2] is used to generate adversarial examples for the perception LEC which cause large error in the distance estimation and result to a collision. It should be noted that although such attacks may be not physically realizable, they provide a framework to analyze the robustness of the LEC. The evaluation results show that the approach can detect such adversarial examples effectively with a short detection delay.

The rest of the paper is organized as follows. Section II introduces system model and formulates the problem. Section III describes the VAE-based regression model and the detection algorithm based in inductive conformal anomaly detection. Section IV presents the evaluation results using the AEBS. Section V concludes the paper.

II. System Model and Problem Formulation

A simplified CPS architecture is shown in Fig. 1. The perception component observes and interprets the environment and feeds the information into the controller which, possibly using additional sensors (feedback from the plant), applies an action to the physical plant in order to achieve some task. In responding to the action, the state of the physical plant changes and the environment is observed and interpreted again to continue the system operation. LECs are extensively used for perception and control tasks in CPS in order to increase the level of autonomy.

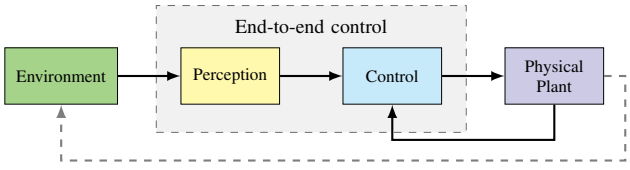


Figure 1. Simplified CPS control architecture [7].

Perception and control may be implemented using LECs (e.g., neural networks). An LEC can be designed using learning techniques such as supervised, unsupervised, and reinforcement learning. It is assumed that the LECs are successfully trained and evaluation of training and testing errors is satisfactory. However, adversarial examples with small input perturbations may cause the LECs to generate outputs with large errors and impact the safety of the system.

The problem considered in this paper is detecting efficiently adversarial examples. During the system operation, the inputs arrive one by one and an adversarial example can cause large prediction error. The objective of the detection is to efficiently detect if the LEC input is adversarial. It should be noted that although such attacks may be not physically realizable, they provide a framework to analyze the robustness of the LEC.

III. Variational Autoencoder for Regression and Adversarial Examples Detection

The detection algorithm proposed in this paper extends the work in [7] by using a VAE-based regression model [12]. The method is based on an LEC architecture which integrates the regression model into the VAE and uses inductive conformal anomaly detection [10].

A. Variational Autoencoder for Regression

A Variational Autoencoder (VAE) is a generative model whose encodings distribution is regularised during the training in order to ensure that its latent space has good properties allowing the generation of new data [13]. The objective is to model the relationship between the observation x and the low-dimensional latent variable z. The architecture presented in [12] integrates a regression model into the generative model to disentangle the regression target variable from the latent space. The architecture of the VAE-based regression model is shown in Fig. 2.

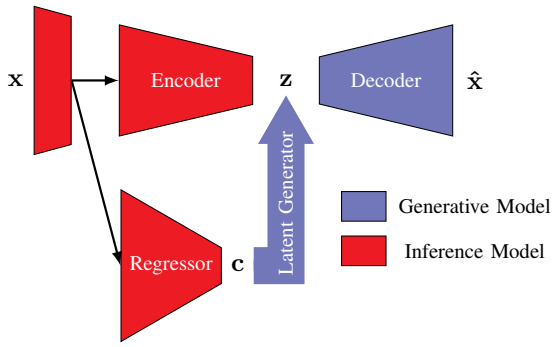


Figure 2. VAE-based regression model [12].

Similar to a traditional VAE, the encoder attempts to represent the input on the latent space using a Gaussian distribution $p(z)$. Instead of using a single Gaussian prior, the VAE-based regression model conditions the prior on the regression target variable c such that the new prior $p(z|c)$, called *latent generator*, can be used to sample a specific latent representation for a given target variable c. On the

other branch of the architecture, a regression network $q(c|x)$ is used to inference the target variable and its uncertainty. The loss function is defined as

$$\begin{aligned}\mathcal{L}(\theta, \phi_c, \phi_z; x) = &- D_{\text{KL}}(q_{\phi_c}(c|x)||p(c)) \\ &+ \mathbb{E}_{z\sim q_{\phi_z}(z|x)}[\log p_\theta(x|z)] \\ &- \mathbb{E}_{c\sim q_{\phi_c}(c|x)}[D_{\text{KL}}(q_{\phi_z}(z|x)||p(z|c))],\end{aligned}$$

where θ, ϕ_z and ϕ_c are the neural network parameters. The first term regularizes the prediction of c with a ground-truth prior $p(c)$. The second term is the model fit trying to reconstruct the input from the latent representations. The third term is the KL divergence between the approximate posterior and the regression-specific prior $p(z|c)$. Both the VAE and the regression model can be jointly trained as a single network and the target variable is disentangled from the latent space.

B. Adversarial Examples for Regression Model

A common approach to generate adversarial examples is the fast gradient sign method (FGSM) [2], which computes an adversarial input by adding a small perturbation in the direction of the gradient. Instead of using this method to make the target network to misclassify the input, in our work, this FGSM is utilized to cause a large error in the output of the regression network.

For a given input x and a target output y^{target}, the neural network output is the regression estimation $c = f(x)$. The objective is to change x to $\tilde{x}$ such that the error between y^{target} and $f(\tilde{x})$ is minimized. This can be achieved by considering the cost function $J(\tilde{x}, y^{\text{target}}) = |f(\tilde{x}) - y^{\text{target}}|^2$ for generating the adversarial examples.

C. Adversarial Detection

1) Conformal prediction: Our method is based on inductive conformal prediction (ICP) [8]. The core idea of the method is to compute a *nonconformity measure* defined by a function A that assigns a value as *nonconformity score* indicating how different a test example from the training data set. A large nonconformity score corresponds to a strange example with respect to the training data set. The original training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ 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 calibration example, the nonconformity scores $\alpha_{m+1}, \ldots, \alpha_l$ are precomputed relative to the proper training set. For a test example (x'_k, y'_k), the nonconformity score α'_k is computed using the same way relative to the proper training set and the p-value is defined as the fraction of calibration examples that have nonconformity scores greater than or equal to the α'_k

$$p_k = \frac{|\{i = m+1, \ldots, l\} \,|\, \alpha_i \geq \alpha'_k|}{l - m}. \tag{1}$$

If the p-value is smaller than a predefined threshold $\epsilon \in (0, 1)$, the test example can be classified as a conformal anomaly. Using a single p-value for detecting the anomaly will make the detector oversensitive. The robustness of the detector can be considerably improved if multiple p-values are used. In [14], it is shown that if the test examples are independent and identically distributed (IID), the p-values are independent and uniformly distributed in $[0, 1]$. Further, a martingale test is used to test the hypothesis that the p-values are independent and uniformly distributed, and the martingale value is used as an indicator for the unusual test example [14]. In [15], the *simple mixture martingale* is used which is defined as

$$M = \int_0^1 \prod_{i=1}^{N} \epsilon p_i^{\epsilon-1} d\epsilon. \tag{2}$$

M will grow only if there are many small p-values in the sequence. More details about the martingales can be found in [15].

The main idea in our approach is to use the generative VAE model to generate multiple examples sampling from the learned latent space probability distribution. The samples which are similar to the input are IID and can be used to generate p-values that are independent and uniformly distributed in $[0, 1]$. Then, the martingale can be used to test this hypothesis and detect if the test example comes from the training data set distribution.

2) Nonconformity measure and adversarial detection algorithm: In [7], the reconstruction error (squared error between the input x and the generated output $\hat{x}$) of the VAE model is used as the nonconformity measure

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

In this paper, we use the same nonconformity measure based on the output of the VAE-based regression model which integrates the regression model and the VAE (Fig. 2).

During the offline phase, the training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ is split into a proper training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ and a calibration set $\{(x_{m+1}, y_{m+1}), \ldots, (x_l, y_l)\}$. The VAE-based regression model is trained by utilizing the proper training data set. For each example in the calibration data set, the nonconformity score is precomputed using the nonconformity measure A_{VAE}.

At runtime, a sequence of test inputs $(x'_1, \ldots, x'_t, \ldots)$ is processed one-by-one. Consider the input x'_t. The method samples from the posterior distribution of the latent space to generate N new examples $\hat{x}'_{t,1}, \ldots, \hat{x}'_{t,N}$, and the nonconformity score $\alpha'_{t,k}$ and the corresponding p-value for each generated example $\hat{x}'_{t,k}$ are computed using A_{VAE} and Eq. (1). Since the generated examples are IID, the p-values $(p_{t,1}, \ldots, p_{t,N})$ are independent and uniformly distributed in $[0, 1]$. Therefore, the martingale in Eq. (2) can be used for detection. The martingale denoted by M_t has a large value if there are many small p-values in the sequence $(p_{t,1}, \ldots, p_{t,N})$ which indicates an out-of-distribution input.

In order to robustly detect when the martingale becomes consistently large, [7] uses a stateful CUSUM detector to generate alarms for out-of-distribution inputs by keeping track of the historical information of the martingale values. The detector is defined as $S_1 = 0$ and $S_t = \max(0, S_{t-1} + M_{t-1} - \delta)$, 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 optimized using empirical data [16]. Typically, after an alarm the test is reset with $S_{t+1} = 0$. Algorithm 1 describes the steps of the method.

Algorithm 1 Adversarial examples detection using the VAE-based regression model

Input: Input training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, test examples $(x'_1, \ldots, x'_t, \ldots)$, number of calibration examples $l - m$, number of examples to be sampled N, stateful detector threshold τ and parameter δ

Output: Output boolean variable $Anom_t$, regression result c_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-based regression model $f(\cdot)$ using the proper training set and get the corresponding nonconformity measure $A_{\text{VAE}}(\cdot)$
for $j = m + 1$ to l **do**
  Sample $\hat{x}_j$ using the trained VAE
  $\alpha_j = A_{\text{VAE}}(x_j, \hat{x}_j)$
end for

Online:

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
  $M_t = \int_0^1 \prod_{k=1}^{N} \epsilon p_{t,k}^{\epsilon-1} d\epsilon$
  if $t = 1$ **then**
    $S_t = 0$
  else
    $S_t = \max(0, S_{t-1} + M_{t-1} - \delta)$
  end if
  $Anom_t \leftarrow S_t > \tau$
  $c_t = f(x'_t)$
end for

IV. Evaluation

We evaluate the adversarial detection method using an advanced emergency braking system (AEBS) implemented using CARLA [11]. The experiments reported use CARLA 0.9.5 on a 16-core i7 desktop with 32 GB RAM and a single RTX 2080 GPU with 8 GB video memory.

A. Experimental Setup

A typical architecture (Fig. 3) and its desirable behavior (Fig. 4) of the AEBS are presented in [7]. The objective of the AEBS is to detect an obstacle and apply appropriate brake force in order to avoid a 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. A perception LEC receives images captured by an on-board camera, detects the nearest front obstacle on the road, and estimates the distance. The estimated distance together with the velocity of the host vehicle are fed to a reinforcement learning controller whose objective is to generate the braking force for safely stopping the vehicle between $L_{\min}$ and $L_{\max}$.

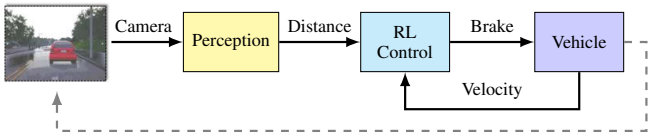


Figure 3. Advanced emergency braking system architecture [7].

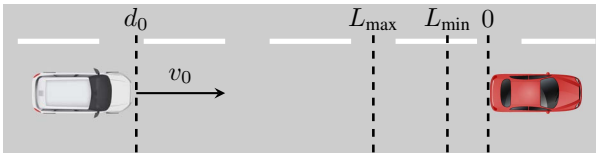


Figure 4. Illustration of advanced emergency braking system [7].

In order to simulate realistic scenarios, the initial velocity v_0 is uniformly sampled between $90\,\text{km/h}$ and $100\,\text{km/h}$, and the initial distance d_0 is based on the camera range in CARLA and is approximately $100\,\text{m}$. The reinforcement learning controller is trained using the DDPG algorithm [17] with 1000 episodes and reward function which aims to stop the vehicle between $L_{\min} = 1\,\text{m}$ and $L_{\max} = 3\,\text{m}$. It should be noted is that the sampling period used in the simulation is $\Delta t = 1/20\,\text{s}$. The reinforcement learning controller is used only for simulation of the closed loop system and does not affect the proposed approach. Details about the design of the reinforcement learning controller can be found in [18].

B. LEC Training

The data set for the perception and detector training consists of 8160 images obtained by varying the initial distance d_0, initial velocity v_0 and precipitation. As described in III, the VAE and the regression network are jointly trained using the VAE-based regression model implemented as a convolutional neural network (CNN).

In the VAE-based regression model, the input firstly passes through four convolutional layers of $32/64/128/256 \times (5 \times 5)$ filters with ELU activations and 2×2 max-pooling, and one fully connected layer of 1568 units with ELU activation. Then, the extracted features are fed into a fully connected layer with 1024 units. The regressor shares the convolutional layers and also has two fully connected layers with $256/1$ units. After the regressor, two fully connected layers of $256/1024$ are used to yield distance-specific latent representations. The decoder has symmetric deconvolutional layers.

A simple two-phase learning schedule is employed with initial searching learning rate $\eta = 10^{-4}$ for 250 epochs, and subsequently fine-tuning $\eta = 10^{-5}$ for 100 epochs. After the training, the mean absolute error of the perception LEC for training and testing are $0.32\,\mathrm{m}$ and $0.40\,\mathrm{m}$ respectively. In addition, multi-dimensional scaling (MDS) [19] is implemented to seek a low-dimensional representation of the latent space of the VAE. Fig. 5 shows the 2D representations of the latent space. From the plots, see that the dimension related to the distance is disentangled from the latent space.

A closed-loop simulation run is shown in Fig. 7. Initially, the distance between the host and the lead car is $98.03\,\mathrm{m}$, and the velocity of the host car is $96.95\,\mathrm{km/h}$ $(= 26.93\,\mathrm{m/s})$. After 140 steps or $7.0\,\mathrm{s}$, the host vehicle stops at $1.83\,\mathrm{m}$ from the lead car.

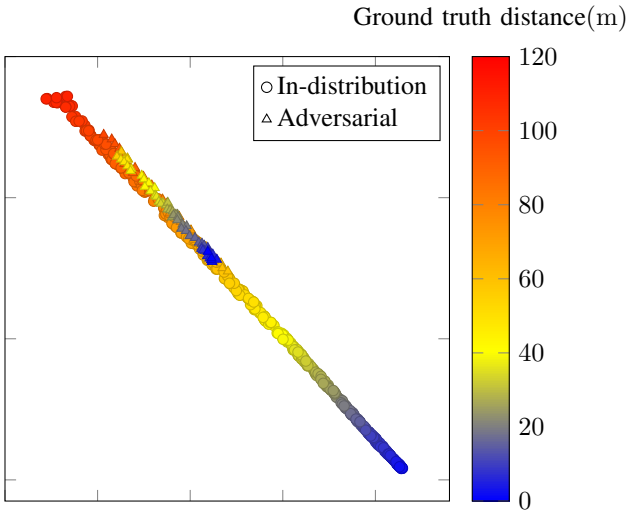
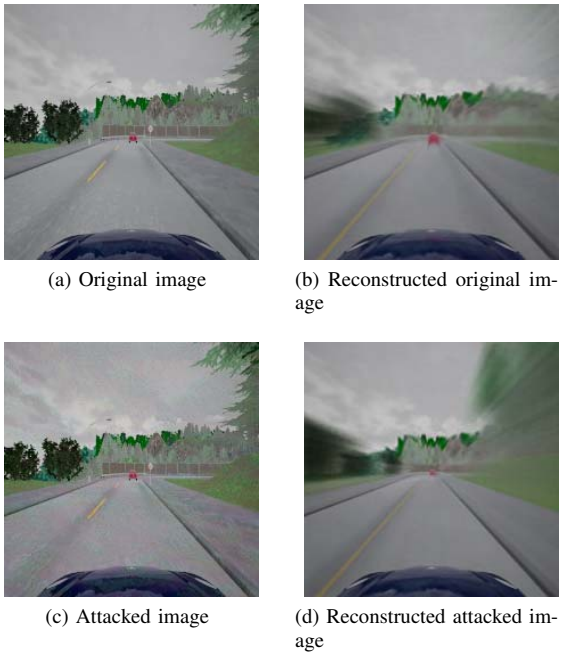


Figure 5. 2D latent representations estimated by VAE-based regression model.

C. Adversarial Examples

The FGSM is used to generate adversarial examples for the perception LEC. For a given input image x_t, the neural network output is the estimated distance $c_t = d_t = f(x_t)$. The objective is to modify x_t by a small step $\epsilon = 0.02$ in the direction that minimizes the loss to $\tilde{x}_t$ such that the predicted distance is close to the target value y^{target}. In our experiment, we set $y^{\text{target}} = 100\,\mathrm{m}$. The perception module predicts a large distance even when the car is very close to a stopped lead car. A comparison of the original image and the adversarial image is shown in Fig. 6a and 6c.



(a) Original image

(b) Reconstructed original image

(c) Attacked image

(d) Reconstructed attacked image

Figure 6. Comparison of original image, image with attack and their reconstructed images.

D. Simulation Results

In order to analyze the performance of the detection method, we generate multiple simulation episodes which contain normal and adversarial examples. Adversarial examples are inserted in the simulation in a time step uniformly sampled from $\{20, 21, \ldots, 60\}$.

To illustrate the approach, we compare two episodes and plot the ground truth and predicted distance to the lead stopped car, the velocity of the host car, the p-values and the S-value of the CUSUM detector (computed using the logarithm of M_t). We use $N = 10$ for the number of the examples generated by the VAE and $\delta = 12, \tau = 80$ for the parameters of the CUSUM detector. Since M_t becomes very large, we choose to use $\log M_t$ to show the results.

The normal case is shown in Fig 7. The p-values are randomly distributed between 0 and 1, and the martingale is small indicating the inputs are in distribution. An episode with adversarial examples is shown in Fig 8. The adversarial example starts at $1.20\,\mathrm{s}$ trying to cause the regression network to predict a larger distance than the actual one. The error starts to increase and reaches almost $20\,\mathrm{m}$. The controller is misled by the perception LEC and does not stop the car in time which collides with the lead car (velocity is

greater than 0 when ground truth distance comes to 0). The p-values become smaller and the detector indicates the test inputs are not from the same distribution as the training data set. The delay for detection is smaller than 10 frames or 0.5 s (the sampling rate is 20 Hz).

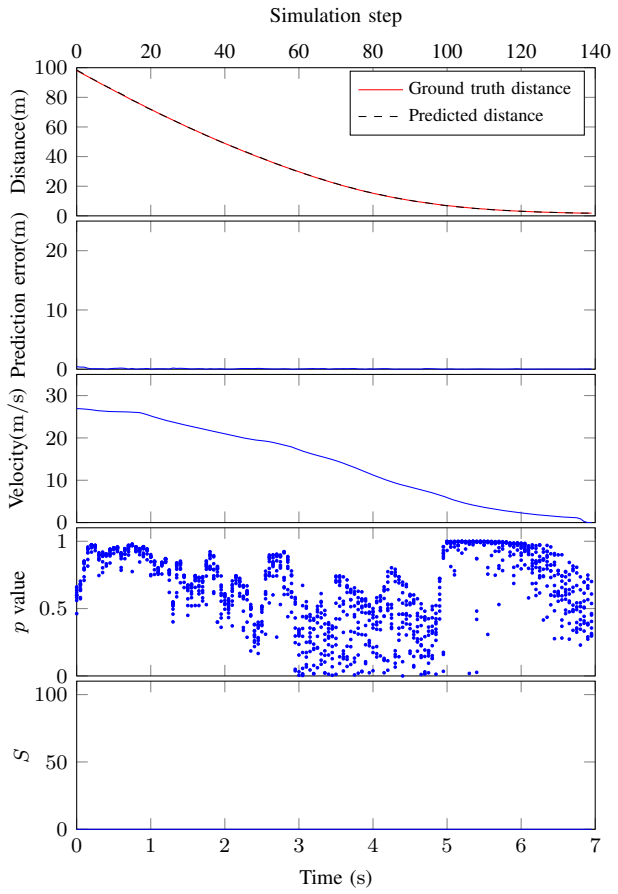


Figure 7. Episode with normal input examples.

We plot the 2D representations of the latent space generated by VAE using one episode of adversarial examples in Fig. 5. Since the latent space is conditioned by the regression prediction, the latent representations of the adversarial examples lie on regions of the space that correspond to different predictions of the regression network. While reconstructing the input for an adversarial example, the VAE generates the new examples from such a region. For example, an image reflecting a small distance is mapped to the region indicating a large distance for an adversarial example. Thus, the reconstructed image is sampled from the wrong region and reflects a large distance, and therefore, the reconstruction error or nonconformity score is large. We compare the original image, adversarial example, and their corresponding reconstructed images in Fig. 6.

We also evaluate the approach for 100 normal episodes and 100 adversarial episodes by considering different values

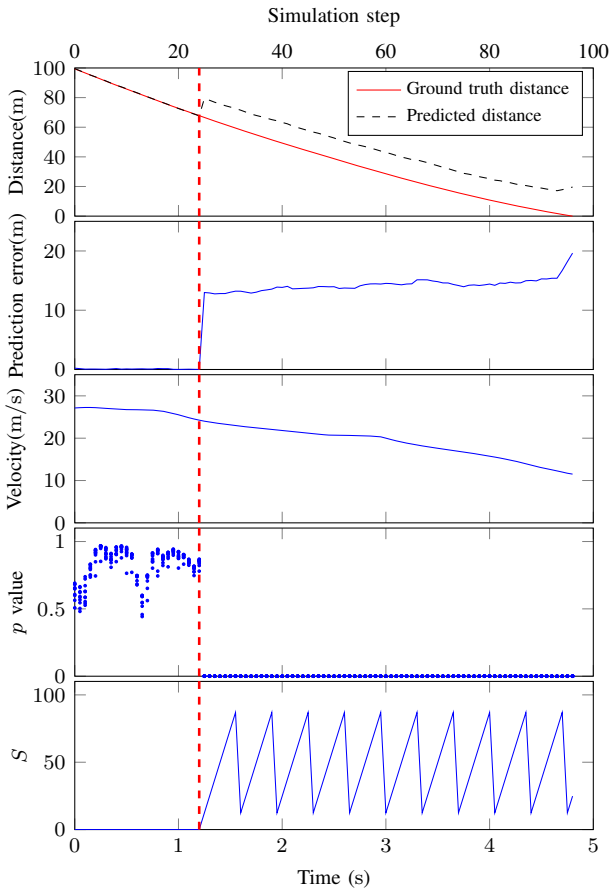


Figure 8. Episode with adversarial examples.

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 adversarial example corresponding to the alarm raised. Table I shows the false alarms and average delay for different N, δ and τ. Since the p-values for the adversarial examples are almost 0, the number of the false alarms is very small and the detection delay is smaller than 10 frames or 0.5 s.

Table I
VAE-BASED DETECTION.

Parameters (N, δ, τ)	False positive	False negative	Average delay (frames)
5, 6, 6	0/100	0/100	1.0
5, 7, 23	0/100	0/100	4.0
10, 10, 62	0/100	0/100	4.0
10, 12, 80	0/100	0/100	6.0
20, 18, 120	0/100	0/100	3.0
20, 20, 280	0/100	0/100	9.0

E. Computational Efficiency

The VAE-based regression model can predict the target variable and compute the nonconformity score in real-time

without storing training data. Table II 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 and (2) the execution times of the VAE-based detectors for different values of N.

Since the VAE-based detection uses N examples in each time step, the execution time is proportional to the number of examples generated N. The execution times are much smaller than the sampling time ($50\,\mathrm{ms}$ in AEBS, and thus, the methods can be used for real-time out-of-distribution detection.

Table II
EXECUTION TIMES.

	N	$\min$ (ms)	Q_1 (ms)	Q_2 (ms)	Q_3 (ms)	$\max$ (ms)
AEBS	N/A	3.49	3.86	3.93	3.98	4.22
VAE	5	18.90	18.96	18.99	19.02	19.08
	10	37.89	37.96	37.99	38.04	38.10
	20	76.32	76.47	76.52	76.71	77.32

V. CONCLUSIONS

In this work, we presented a detection method for adversarial examples in learning-enabled CPS. The method is based on inductive conformal prediction and uses a VAE-based regression model to predict the target variable and compute the nonconformity of new inputs relative to the training set. The evaluation is based on an AEBS implemented in an open source simulator for self-driving cars. FGSM is used to generate adversarial examples. The results demonstrate that the method can efficiently detect adversarial examples with a short detection delay. Future work includes detection of physically realizable attacks, comparing this new approach with other adversarial detection methods to study into the benifit of taking the output into consideration, and also investigating the generation of adversarial examples that are not detectable by the approach.

REFERENCES

[1] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.

[3] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal, "Darts: Deceiving autonomous cars with toxic signs," 2018.

[4] D. Meng and H. Chen, "Magnet: A two-pronged defense against adversarial examples," in *ACM Conference on Computer and Communications Security*. ACM, 2017, pp. 135–147.

[5] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE, (2)1*, 2015.

[6] L. Ruff, R. A. Vandermeulen, N. Görnitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, 2018, pp. 4393–4402.

[7] F. Cai and X. Koutsoukos, "Real-time out-of-distribution detection in learning-enabled cyber-physical systems," 2020.

[8] V. Balasubramanian, S.-S. Ho, and V. Vovk, *Conformal prediction for reliable machine learning: theory, adaptations and applications*. Newnes, 2014.

[9] D. Volkhonskiy, E. Burnaev, I. Nouretdinov, A. Gammerman, and V. Vovk, "Inductive conformal martingales for change-point detection," *Proceedings of Machine Learning Research*, vol. 60, pp. 1–22, 2017.

[10] R. Laxhammar and G. Falkman, "Inductive conformal anomaly detection for sequential detection of anomalous sub-trajectories," *Annals of Mathematics and Artificial Intelligence*, vol. 74, no. 1-2, pp. 67–94, 2015.

[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[12] Q. Zhao, E. Adeli, N. Honnorat, T. Leng, and K. M. Pohl, "Variational autoencoder for regression: Application to brain aging analysis," *arXiv preprint arXiv:1904.05948*, 2019.

[13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.

[14] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

[15] V. Fedorova, A. Gammerman, I. Nouretdinov, and V. Vovk, "Plug-in martingales for testing exchangeability on-line," in *29th International Conference on Machine Learning*, 2012, pp. 923–930.

[16] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application*. Prentice Hall Englewood Cliffs, 1993, vol. 104.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *International Conference on Learning Representations*, 2016.

[18] H. Tran, F. Cai, D. M. Lopez, P. Musau, T. T. Johnson, and X. D. Koutsoukos, "Safety verification of cyber-physical systems with reinforcement learning control," *ACM Trans. Embedded Comput. Syst.*, vol. 18, no. 5s, pp. 105:1–105:22, 2019.

[19] I. Borg and P. Groenen, "Modern multidimensional scaling: Theory and applications," *Journal of Educational Measurement*, vol. 40, no. 3, pp. 277–280, 2003.