Self-Secured Control with Anomaly Detection and Recovery in Automotive Cyber-Physical Systems

Korosh Vatanparvar, Mohammad Abdullah Al Faruque
Henry Samueli School of Engineering, EECS Department
University of California, Irvine
Irvine, California, USA
{kvatanpa, alfaruqu}@uci.edu

***Abstract*—Cyber-Physical Systems (CPS) are growing with added complexity and functionality. Multidisciplinary interactions with physical systems are the major keys to CPS. However, sensors, actuators, controllers, and wireless communications are prone to attacks that compromise the system. Machine learning models have been utilized in controllers of automotive to learn, estimate, and provide the required intelligence in the control process. However, their estimation is also vulnerable to the attacks from physical or cyber domains. They have shown unreliable predictions against unknown biases resulted from the modeling. In this paper, we propose a novel control design using conditional generative adversarial networks that will enable a self-secured controller to capture the normal behavior of the control loop and the physical system, detect the anomaly, and recover from them. We experimented our novel control design on a self-secured BMS by driving a Nissan Leaf S on standard driving cycles while under various attacks. The performance of the design has been compared to the state-of-the-art; the self-secured BMS could detect the attacks with 83% accuracy and the recovery estimation error of 21% on average, which have improved by 28% and 8%, respectively.**

***Index Terms*—CPS, Security, Electric Vehicle, Battery, Machine Learning, Battery Management System**

I. Introduction and Related Work

Emerging technologies and introduction of Cyber-Physical Systems (CPS) have initiated new domains of applications that are more complex and intelligent. Advancement of technology and fabrication process have enabled production of smaller sensors and actuators that can facilitate implementation of complex tasks [1–4]. For instance, new powerful sensors, computing resources have enabled advancement in automotive industry such as production of Electric Vehicles (EV) and implementing Advanced Driver-Assistance Systems (ADAS) to enable different levels of autonomous driving [5, 6].

Electronic Design Automation (EDA) has helped with addressing challenges towards design and development of the CPS. Controllers are integrated within these systems that comprise of: sensors to monitor the environment and state of the physical system; micro controllers to process the data and make decisions; and actuators to apply the control actions to the physical system [7]. Sophisticated control algorithms are implemented that make decisions based on phsyical state and given a complex model. For instance, a Battery Management System (BMS) utilizes a model of the battery in order to decide how much each cell can provide power and energy at each state. The BMS is responsible to prevent over loading, over charging, and over discharging the battery cells considering power requests received from the system, e.g. EV. Moreover, resource management and scheduling algorithms may be implemented using Model Predictive Control (MPC) or Reinforced Learning to distribute the power among the battery cells or even other types of energy storage such as ultracapacitor [8, 9]. Hence, the main objective of the BMS is to improve the available capacity and energy efficiency while minimizing the battery lifetime degradation that may happen in lithium-ion battery cells [10, 11].

Decisions made at run time by the controller depend on the observed state of the physical system and control inputs are adjusted based on the model. However, the arising challenge is that whether the controller can trust the sensed data and/or the model. There are many scenarios that the control loop can be compromised. The corrupted loop will cause the controller to make wrong decisions and thereby sway the state of the physical system to unwanted or unstable states [12–14].

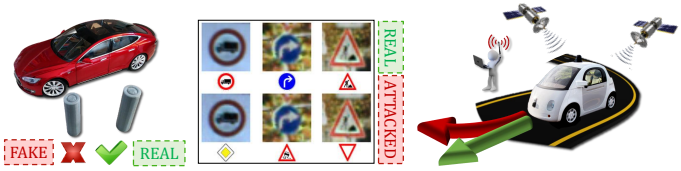

Fig. 1. Examples of Different Existing Attacks on the Control in a Compromised CPS. [15]

A) Motivational Example of a Compromised System

Control loops can be compromised from different aspects such as data sensing, the physical system, or the model (see Figure 1). For instance, the sensed data can become unreliable by directly or indirectly attacking the sensors to give wrong data, no data, or misplaced data (e.g. Denial-of-Service or Delay Attacks). Example of this attack has been seen in navigation systems by altering the data received from the GPS or IMU sensors [16, 17]. Moreover, the physical system which is the main module of the control loop can be attacked; a battery may be replaced with a low-quality alternative and cause the whole CPS to catch on fire since the BMS is unaware of the alteration of the physical system. Furthermore, biased machine learned models in the controllers have been seen to give wrong decisions by changing the control inputs slightly not visible to naked eyes. For instance, an image classifier model in an autonomous driving control may detect a "STOP" sign as a "Speed Limit" sign [18].

This material is based upon work partially supported by the National Science Foundation under Grant No. ECCS 1611349 and the University of California, Office of the President under Grant No. LFR-18-548175.

B) Summary and Conclusion from Observation

According to the observation and experiments, the state-of-the-art controllers are not intelligent enough to capture anomaly in the control loop. In other words, the controller makes the decision based on the current observed state of the physical system and the given model without considering that the behavior of the control loop is not normal. Moreover, the controller does not have any mechanism to secure the control loop to prevent a compromise and to recover from the attack.

C) Our Novel Contributions

In order to address the challenges with vulnerabilities existing in control, we propose a novel self-secured machine learning architecture by employing the following.

1) **Control Loop Vulnerabilities (Section II):** are described in details for the current controllers of the automotive CPS. Different aspects that the control loop can be compromised and our solution to the issues are explained.
2) **Self-Secured Machine Learning (Section III):** architecture is proposed that utilizes a novel Conditional Generative Adversarial Network (CGAN) to capture the behavior of the control loop and detect any anomaly at the run time [19]. Moreover, a novel secure prediction technique will recover the control loop from the attack.
3) **Self-Secured BMS (Section IV):** is implemented where the self-secured machine learning architecture is integrated into an existing BMS and tested against multiple existing attack models.

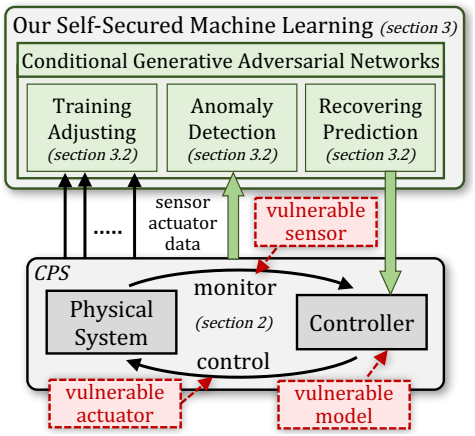

Fig. 2. Our contributions towards a self-secured machine learning architecture integrated into automotive CPS.

Figure 2 illustrates the highlighted modules of the self-secured machine learning architecture that will be integrated into an existing automotive CPS, e.g. battery management system. Moreover, it abstracts how the modules are related to each other to detect a vulnerability and recover from it.

II. Control Loop Vulnerabilities

Cyber-physical systems typically implement controllers that are responsible for interacting with one or more physical systems to reach an objective and maintain a certain control quality [20]. In other words, controller seeks to maintain physical variables at certain set points in spite of unmeasured disturbances. The interaction between the physical systems and the controller is abstracted as a closed-loop feedback control.

A. Control Loop Design

Control loops comprise multiple components of sensors, microcontrollers, and actuators, interacting with a physical system (see Figure 3).

Sensors are devices that measure and convert certain types of energy in terms of a physical parameter to an electrical output. Sensors are used to monitor the state of a physical system or environment. For instance, a BMS may implement thousands of sensors on a battery module attached to each cell to measure their voltage, current, or temperature [8].

Microcontrollers periodically retrieve the data from the sensors and process them to make a decision for controllable variables. Typically, Proportional Integral Derivative (PID) controllers are responsible to accomplish such task. However, PIDs have very limited knowledge of the physical system and its dynamic behavior. Hence, Model Predictive Control (MPC) is a more advanced method of control that relies on dynamic models of the physical system. They achieve better control stability and quality of control by optimizing the control process for a certain time horizon in the future. For instance, the BMS may process the data to evaluate the battery cell State-of-Charge (SoC) or battery lifetime - State-of-Health (SoH) [21]. The BMS will then find the optimal control actions considering the battery lifetime and energy efficiency.

Actuators are devices that convert an electrical signal to the required physical parameter, e.g. current, voltage, physical motion, etc. The electrical signal to the actuators are decided by the microcontrollers. For instance, the BMS may adjust the current drawn from the battery cells or activate relay switches to change the battery structure as part of control actions resulted from the BMS algorithm [22].

Physical System dynamics will change by triggering the actuators and applying the control actions decided by the microcontroller. The control process will continue periodically to maintain the required set points for the physical system. For instance, due to the intrinsic variations in battery cells, they may discharge differently from each other. Hence, the BMS observing the state of these cells will adjust its controllable variables (current drawn from the cells) to maintain the balance among them and increase the available battery capacity.

B. Physical System Attack

Physical system dynamics is typically described using mathematical equations such as Ordinary Differential Equations (ODE) or Finite-State Machines (FSM). These models can be used in MPC algorithms for estimating the state of the system and evaluating optimum control actions.

Battery cells have different performances and illustrate different current-voltage (I-V) and thermal behavior during charge and discharge cycles due to the various materials used in the production. High performance battery cells are typically required for high-power or high-energy applications. However, they cost much more than the low performance ones. Hence, middle-man-attack may replace or alter the cells with the counterfeit ones without the BMS being aware of the alteration. The BMS interacting with the new counterfeit physical system may not operate properly resulting in unstable

states, e.g. fast draining battery cells or cells catching on fire [23]. Currently, there exist multiple mechanisms to distinguish between these cells. For instance, the appearance and packing can be used to tell the difference or an embedded RFID tags can be used. However, the current approaches can be hacked easily leaving the control loop vulnerable.

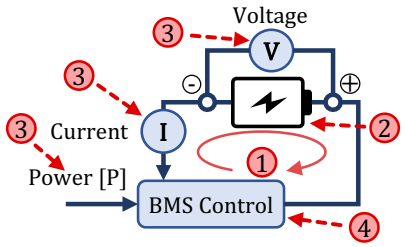

Fig. 3. Control loop design (1) in BMS and its vulnerabilities in automotive CPS: physical attack (2), sensor attack (3), and vulnerable model (4).

C. Sensor Attack

Nowadays, to maintain scalability of the sensor network with the growing number of sensors implemented in CPS, the connections are becoming wireless, e.g. Bluetooth, Zigbee. The wireless network can bring more weak points and add more vulnerabilities to a control loop. Therefore, the sensor data can be under multiple attacks, e.g. man-in-the-middle, fuzz attack, replay attack. Therefore, the controller may not be able to trust the data for observing the state and making decisions. For instance, a compromised voltage sensor of a battery may cause the BMS to over charge or over discharge the battery resulting in shorter battery lifetime or in the worst case explosion. Cryptography algorithms are typically used to secure the communication channel for data. However, they have their own challenges in terms of complexity scalability. Furthermore, they will not be effective against physical attacks to the sensors, leaving the control loop vulnerable.

D. Vulnerable Model

Physical system models utilized by the algorithms will help the controller identify the current state of the physical system given the observed data. For instance, an object detection algorithm classifies the recorded image from the camera based on a pre-trained machine learning model. The BMS estimates the SoC of the battery cell given the measured current drawn from the battery. However, the reliability of the model and decisions made by the controller are unknown without testing. In other words, the attacker can remain in a stealth mode wherein it spoofs the sensors to an extent that is indistinguishable from noise. However, the attacker can force the system to get into an unsafe region [24]. For instance, the object detection algorithm can be fooled to classify an image completely wrong with the highest probability by slightly tweaking the image (not visible to naked eye) [15]. On the other hand, the BMS may be attacked to consume more energy from the battery cells [25].

III. Self-Secured Control

We explained the security and vulnerability challenges with the current controls. Hence, in this paper, we propose a novel machine learning architecture using Conditional Generative Adversarial Network (CGAN) that will be integrated in parallel to the control loop (see Section III-A). It is responsible to capture and learn the normal behavior of the physical system interacting with the controller (see Section III-B). The architecture is trained by running the CPS and monitoring the control loop at run time by the manufacturer, before any attack can happen, at ***train-only*** phase. Afterwards, at ***detect-n-predict*** phase, the architecture will monitor the control loop to detect any anomaly or attack (see Section III-C) and recover from them (see Section III-D) at run time. Moreover, it will also get updated and learn new dynamics, if the rate of attacks detected are low in a certain time window.

A. Machine Learning Architecture

Typically, the dynamic behavior of the control loop containing the physical system and the control components can be modeled using mathematical equations and deterministic modeling. However, the behavior of the physical systems can get too complex to be modeled by equations. Moreover, there may be many unknown factors influencing the behavior that make it challenging to model. For instance, a battery manufacturing process is not completely deterministic that will result in various battery cells with different performance and behavior. Modeling and capturing the exact behavior of the battery cells is very challenging problem. Hence, data-driven statistical modeling (machine learning) is applied to describe the behavior. However, machine learning models may suffer from unknown biases and sometimes significant prediction errors. Therefore, we propose a novel conditional generative adversarial network to capture the behavior. Moreover, it will also help in enabling a self-secured control.

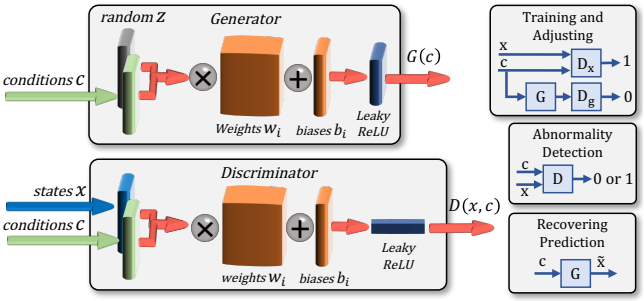

Fig. 4. Our novel conditional generative adversarial network architecture for self-secured control.

Generative Adversarial Networks (GAN) adapts a machine learning architecture to help with generating a more stable model [19]. By the definition of GAN, there will be two neural networks 1) *generator* (G) and 2) *discriminator* (D) (see Figure 4). GAN creates a situation for both neural networks that can be modeled as a minimax game in game theory. Hence, for the *generator* to be successful, it needs to learn to generate the distribution of the real data for the physical process very well, such that the *discriminator* cannot distinguish. On other hand, for the *discriminator* to be successful, it needs to learn the distribution of the real data for the physical process very well, such that the *generator* cannot fool it. Therefore, in order to compete with each other to get better in this game, both will become the best to generate

and discriminate. Hence, at the equilibrium point, which is the optimal point in minimax game, the *generator* will model the real data, and the *discriminator* will output probability of 0.5 as the output of the *generator* equals real data. There are many variations to the architecture and loss function of GAN which are similarly applicable to the proposed control and their performance difference is out of the scope of the paper.

Selection of GAN architecture for the control design would be mainly due to the following factors: 1) competition between two networks can provide faster convergence; 2) two networks are already being trained for purposes required for a secured control design; 3) the model will be more robust towards any attack model especially adversarial examples without the need for application-specific data.

Here in this paper, both neural networks need to learn the physical process of the control loop for a limited number of time steps ($T = 6$). The data is sampled from the signals of the sensors and actuators in the control loop. The *discriminator* verifies whether the physical process data for the period of T is real or fake (compromised). On the other hand, the *generator* attempts to generate similar data to fool the discriminator.

To capture the current state of the physical system and make the decisions more aware of the context, a Conditional GAN (CGAN) is implemented. Here in, the beginning portion of the time steps ($T_c = 4 < T$) is given as the condition data to both networks. Therefore, both networks will predict based on the given condition. Furthermore, the conditional prediction will help the *generator* to later do recovery prediction that will be discussed in Section III-D.

B. Training and Adjusting

The main challenge of the CGAN architecture is training two neural networks. Both networks need to converge to an equilibrium point where none of them are too much stronger than the other. Otherwise, the *discriminator* always tells the difference between the real or fake data or the *generator* always generates very closely to real data.

Hence, at the ***train-only*** phase, the manufacturer is responsible to run the controller for a certain period of time in order to train the CGAN with real data. During the training process, two optimizations will be conducted to minimize their loss functions.

$$G_{loss} = \frac{1}{N}\sum_{i=1}^{N} H\left(D\left(G\left(c_i\right), c_i\right), 1\right) \tag{1}$$

where G_{loss} loss function is to minimize the mean of the cross entropy between label one (1) and the output of the *discriminator* given the fake generated data based on condition c_i over N batch size. In other words, training the *generator* to fool the *discriminator*.

$$D_{loss}^{real} = \frac{1}{N}\sum_{i=1}^{N} H\left(D\left(x_i, c_i\right), 1\right) \tag{2}$$

$$D_{loss}^{fake} = \frac{1}{N}\sum_{i=1}^{N} H\left(D\left(G\left(c_i\right), c_i\right), 0\right) \tag{3}$$

$$D_{loss} = D_{loss}^{real} + D_{loss}^{fake} \tag{4}$$

D_{loss} loss function is to minimize the mean of the cross entropy between label zero (0) and the output of the *discriminator* given the real data and condition c_i, plus the cross entropy between label one (1) and the output of the *discriminator* given the fake generated data based on condition c_i.

After training the CGAN, both entropies will reach small stable values. However, the training will not stop after the ***train-only*** phase. The CGAN will be further trained at the ***detect-n-predict*** phase given the new batches of data sampled at run time. However, only the consecutive batches are used for training that their probability of having no anomaly is higher than an arbitrary defined trust threshold (α_{real}) for a large period of time. Meanwhile, at the ***detect-n-predict*** phase, the CGAN is used for anomaly detection and recovering prediction as will be discussed in the following.

C. Anomaly Detection

The CGAN *discriminator* captures the real dynamic behavior of the control loop for T time steps given the conditional data for T_c time steps. Due to competition with the *generator*, it does not get adapted to fake or corrupted data and it is less biased and more tolerant to adversarial attacks.

Any attack to the vulnerable physical system, sensor, or model will corrupt the physical process of the control loop (see Section II). Therefore, the *discriminator* can give a probability of detecting an anomaly in the period of the given data x and condition c. The conditional anomaly detection helps in capturing the state of the physical system for the last T_c time steps to make more deterministic decision. When the probability of the given batch being normal is smaller than a fake threshold $[D\left(x, c\right) < \alpha_{fake}]$, the batch will be labeled as an anomaly. In other words, the control loop is compromised.

Many defensive mechanisms can be implemented in case of detecting an anomaly, e.g. triggering default control actions. However, we introduce a recovering prediction mechanism that can complement or replace the current mechanisms.

D. Recovering Prediction

Trained CGAN includes a *generator* neural networks that is used for generating data (fake) resembling the real data of the physical process. The trained *generator* captures the real dynamic behavior of the control loop for T time steps given the conditional data for T_c time steps. Therefore, using the *generator* model, the dynamic behavior of the physical system can be predicted, when needed.

When an anomaly is detected, the current conditional data can be given to the *generator* network to generate the rest of the physical data. Although the data is fake, it is very close to the real data that might have happened when there was no anomaly. The first estimated physical data: $\tilde{x}[T_c + 1] \leftarrow \tilde{x} = G\left(c\right)$ will be applied to the control loop.

IV. Self-Secured BMS

We apply our novel CGAN machine learning architecture to an existing battery management system to enable self-security against various attacks and vulnerabilities.

A. Attack Models

Multiple attack models are applied to the control loop of a BMS to observe physical system behavior change.

a) Physical Attack: in this attack model, the physical system is altered, for instance the battery cell is replaced with lower performance battery cell. The internal resistance and capacitance of the battery cell will be higher. The behavior is not detectable by normal BMS since the cell may still provide enough power required

b) Denial-of-Service Attack: the data retrieved from the sensors are tampered with uniform distribution noise that is randomly generated with the probability of 20%. This attack will directly fool the controller to observe a wrong (random) state of the system, e.g. higher SoC or lower voltage than actually available.

B. Integrating Novel Architecture

We apply our CGAN architecture to an existing BMS. Voltage, current, and power sensor values are sampled every second. They will represent the dynamic behavior of the control loop (physical process) - the I-V characteristics of a battery cell. Moreover, the relationships between these data values should follow the correct behavior of the control loop (without a compromised control or sensor).

V. Experimental Results

To test the functionality and performance of our self-secured control using CGAN, the self-secured BMS is experimented on multiple attack scenarios and the performance is compared with a normal BMS and single generator trained individually for both anomaly detection and recovery prediction.

A. Experiment Setup

The battery, sensors, and actuators are modeled in MATLAB. Lithium-ion battery cell 18650 has been used for the experiment. A Nissan Leaf S EV has been driven on a standard driving cycle *NEDC* and *ECE* [8, 10] as case studies. The training and prediction of the CGAN machine learning model has been implemented using TensorFlow in python [26]. Hence, the control algorithms of the self-secured BMS is running in python and communicating with MATLAB to retrieve sensor data and transmit control actions.

B. Results and Analysis

We analyze and compare the performance of the self-secured BMS in terms of model accuracy of CGAN for training data, anomaly detection performance of the *discriminator* networks, and estimation error of the *generator* network in prediction.

a) CGAN Model Accuracy: at the ***train-only*** phase, the CGAN model is trained on 10,000 samples of data by driving the Nissan Leaf S EV on a standard driving cycle *NEDC* and *ECE*. The loss function of the both networks are shown in Figure 5. It is shown that after a while, the value of cross entropy for both loss functions reaches a stable value, when they have captured the physical process thoroughly. There might be new data samples at run time that the models have not captured and there should be a peak in the error. However, these will not be detected as anomaly since they are higher than than the fake threshold.

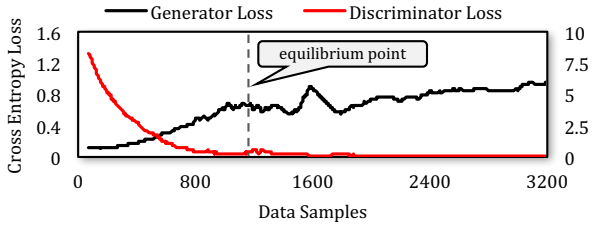

Fig. 5. CGAN *generator* and *discriminator* loss function at *train-only* phase reaching equilibrium point.

The performance shown in this paper is the result after tuning the CGAN machine learning networks by optimizing multiple hyper parameters. The size of the inputs the networks are $T_c = 4$ and $T - T_c = 2$ multiplied by the number of sensors (3). The *discriminator* has one hidden layer with 360 neurons and the *generator* has one hidden layer with 1200 neurons. The dimension of added noise in the *generator* is 15. These parameters are adjusted such that the models do not over fit the data and can reach equilibrium point.

b) Anomaly Detection: at the ***detect-n-predict*** phase, we apply two attack models (see Section IV-A) and observe how the self-secured BMS detects them. Figure 6 illustrates the metric that the discriminator predicts the behavior to be correct. When there is no attack it is in a normal range. However, when the behavior is corrupted in a compromised control, the values fall below the threshold as shown in the figure. The threshold is defined based on the minimum value achieved in the ***train-only*** phase. The denial-of-service attack happens 20% of the time with a uniform distribution. It is shown that 83% of the attacks are detected. The physical attack is much harder since the behavior is complex to capture. However, our self-secured control is able to detect the attack by identifying the fake behavior. It has been seen that the single generator technique would not be able to detect more 65% of the attacks. Since the decision is mainly based on the error of the prediction and current state, it does not capture behavior change when there is anomaly.

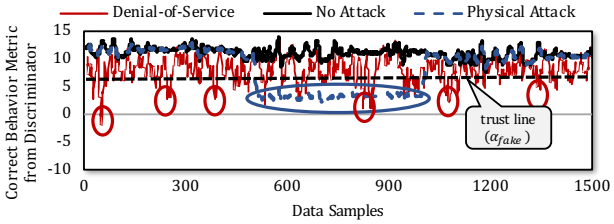

Fig. 6. Performance of the *discriminator* in detecting DoS and physical attack.

c) Prediction Recovering Error: at the ***detect-n-predict*** phase, we apply two attack models (see Section IV-A) and observe how the self-secured BMS recovers from the detected anomalies by predicting. The error of the prediction values from the *generator* is compared with the real values of the physical system at the run-time ***detect-n-predict*** phase. Figure 7 shows the probability density of the prediction error. As shown in the figure, the estimation error when a physical

attack happens, is more deterministic than DoS. This is due to the fact that the physical behavior is compromising the system in a more deterministic way than a random number jamming a data (DoS). The average prediction error resulted from the *generator* is about 21% which is significantly good for a model which has no prior knowledge of the system. The error has decreased compared to a single generator which was 29%.

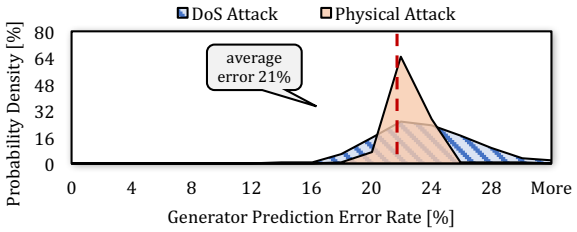

Fig. 7. Performance of the *generator* in terms of the prediction error when an attack happens.

VI. Conclusions

CPS are vulnerable to attacks from sensors/actuators, unknown biases, and model predictions. Current security solutions such as cryptography do not address these attacks from the physical domain. Hence, in this paper, we have proposed a machine learning architecture using CGAN to enable a self-secured control. CGAN will capture the dynamic behavior of the control loop in order to detect any anomaly resulted from the attacks, and to recover from the attack by predicting the correct state of system. We experimented a self-secured BMS by driving a Nissan Leaf S on *NEDC*, *ECE* driving cycles. The self-secured BMS could detect the added attacks to the system with 83% accuracy and the recovering prediction error has been 21% on average which improved by 28% and 8%, respectively.

References

[1] Andy Heinig, Manfred Dietrich, et al. System Integration - The Bridge between More than Moore and More Moore. *Proceedings of the Conference on Design, Automation & Test in Europe*, page 132, 2014.

[2] Marco Casale-Rossi, Pietro Palella, Mario Anton, et al. The World Is Going... Analog & Mixed-Signal! What about EDA? *Proceedings of the Conference on Design, Automation & Test in Europe*, page 37, 2014.

[3] Korosh Vatanparvar and Mohammad Abdullah Al Faruque. ACQUA: Adaptive and Cooperative Quality-Aware Control for Automotive Cyber-Physical Systems. *International Conference on Computer-Aided Design (ICCAD)*, pages 1–8, 2017.

[4] Korosh Vatanparvar, Sina Faezi, et al. Extended Range Electric Vehicle with Driving Behavior Estimation in Energy Management. *IEEE Transactions on Smart Grid*, 2018.

[5] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2722–2730, 2015.

[6] Maral Amir and Tony Givargis. Priority Neuron: A Resource-Aware Neural Network for Cyber-Physical Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2732–2742, 2018.

[7] Ragunathan Raj Rajkumar, Insup Lee, et al. Cyber-Physical Systems: The Next Computing Revolution. *47th Design Automation Conference (DAC)*, pages 731–736, 2010.

[8] Sangyoung Park, Younghyun Kim, and Naehyuck Chang. Hybrid Energy Storage Systems and Battery Management for Electric Vehicles. *50th Design Automation Conference (DAC)*, pages 1–6, 2013.

[9] Korosh Vatanparvar and Mohammad Abdullah Al Faruque. OTEM: Optimized Thermal and Energy Management for Hybrid Electrical Energy Storage in Electric Vehicles. *Conference on Design, Automation & Test in Europe (DATE)*, pages 1–6, 2016.

[10] Korosh Vatanparvar and Mohammad Abdullah Al Faruque. Design and Analysis of Battery-Aware Automotive Climate Control for Electric Vehicles. *ACM Transactions on Embedded Computing Systems (TECS)*, 2018.

[11] Donghwa Shin, Massimo Poncino, et al. A Statistical Model-Based Cell-to-Cell Variability Management of Li-ion Battery Pack. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(2):252–265, 2015.

[12] Hamza Fawzi, Paulo Tabuada, and Suhas Diggavi. Secure Estimation and Control for Cyber-Physical Systems Under Adversarial Attacks. *IEEE Transactions on Automatic Control*, 59(6):1454–1467, 2014.

[13] Anthony Bahadir Lopez, Korosh Vatanparvar, et al. A Security Perspective on Battery Systems of the Internet of Things. *Journal of Hardware and Systems Security*, 1(2):188–199, 2017.

[14] Armin Wasicek, Patricia Derler, et al. Aspect-oriented Modeling of Attacks in Automotive Cyber-Physical Systems. *51st Design Automation Conference (DAC)*, pages 1–6, 2014.

[15] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[16] Alan Grant, Paul Williams, Nick Ward, and Sally Basker. GPS jamming and the impact on maritime navigation. *The Journal of Navigation*, 62(2):173–187, 2009.

[17] Timothy Trippel, Ofir Weisse, Wenyuan Xu, et al. Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks. *IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 3–18, 2017.

[18] Nilaksh Das, Madhuri Shanbhogue, et al. Keeping the Bad Guys Out: Protecting and Vaccinating Deep Learning with JPEG Compression. *arXiv preprint arXiv:1705.02900*, 2017.

[19] Ian Goodfellow, Jean Pouget-Abadie, et al. Generative Adversarial Nets. *Advances in neural information processing systems*, pages 2672–2680, 2014.

[20] Steffen Lampke, Simon Schliecker, Dirk Ziegenbein, and Arne Hamann. Resource-Aware Control-Model-Based Co-Engineering of Control Algorithms and Real-Time Systems. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, 8(2015-01-0168):106–114, 2015.

[21] Ho-Ta Lin, Tsorng-Juu Liang, et al. Estimation of Battery State of Health Using Probabilistic Neural Network. *IEEE Transactions on Industrial Informatics*, 9(2):679–685, 2013.

[22] Liang He, Yu Gu, et al. SHARE: SoH-Aware Reconfiguration to Enhance Deliverable Capacity of Large-Scale Battery Packs. *International Conference on Cyber-Physical Systems (ICCPS)*, pages 169–178, 2015.

[23] Qingsong Wang, Ping Ping, et al. Thermal runaway caused fire and explosion of lithium ion battery. *Journal of power sources*, 208:210–224, 2012.

[24] Yilin Mo and Bruno Sinopoli. On the Performance Degradation of Cyber-Physical Systems Under Stealthy Integrity Attacks. *IEEE Transactions on Automatic Control*, 61(9):2618–2624, 2016.

[25] Thomas Martin, Michael Hsiao, Dong Ha, and Jayan Krishnaswami. Denial-of-Service Attacks on Battery-powered Mobile Computers. *Pervasive Computing and Communications (PerCom)*, pages 309–318, 2004.

[26] Martín Abadi, Ashish Agarwal, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.