ROLDEF: RObust Layered DEFense for Intrusion Detection Against Adversarial Attacks

Onat Gungor[1,2], Tajana Rosing[1], and Baris Aksanli[2]

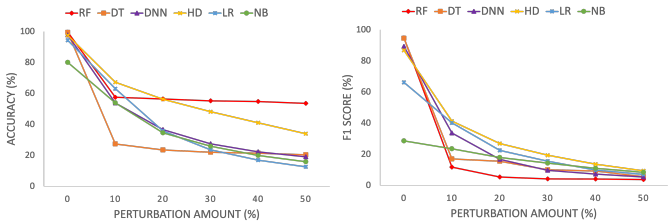
[1]Department of Electrical and Computer Engineering, University of California, San Diego
[2]Department of Electrical and Computer Engineering, San Diego State University
ogungor@ucsd.edu, tajana@ucsd.edu, baksanli@sdsu.edu

***Abstract*—The Industrial Internet of Things (IIoT) includes networking equipment and smart devices to collect and analyze data from industrial operations. However, IIoT security is challenging due to its increased inter-connectivity and large attack surface. Machine learning (ML)-based intrusion detection system (IDS) is an IIoT security measure that aims to detect and respond to malicious traffic by using ML models. However, these methods are susceptible to adversarial attacks. In this paper, we propose a RObust Layered DEFense (*ROLDEF*) against adversarial attacks. Our denoising autoencoder (DAE) based defense approach first detects if a sample comes from an adversarial attack. If an attack is detected, adversarial component is eliminated using the most effective DAE and the purified data is provided to the ML model. We use a realistic IIoT intrusion data set to validate the effectiveness of our defense across various ML models, where we improve the average prediction performance by 114% with respect to no defense. Our defense also provides 50% average prediction performance improvement compared to the state-of-the-art defense under various adversarial attacks. Our defense can also be deployed for any underlying ML model and provides an effective protection against adversarial attacks.**

I. Introduction

The Industrial Internet of Things (IIoT) is the connection of industrial assets with the information systems and the business processes [1]. It continuously monitors and analyzes collected data towards better system efficiency and reliability. Its value increases where IIoT could be worth $7.1 trillion in the United States by 2030 [2]. Increased inter-connectivity and poorly implemented security features make IIoT an easy target for cybercriminals [3]. An adversary can exploit vulnerabilities to modify system data, disrupt communication, or prevent asset availability [4]. Recent cyberattacks include StuxNet and Industroyer [3]. IIoT security is challenging due to long lifetime of industrial devices and increased interconnection among IIoT devices. Intrusion Detection System (IDS) is a security solution that continuously monitors the network data to detect cyberattacks [5]. Machine learning (ML) methods have been recently adopted for IDS due to their accuracy [6].

Although ML methods provide good intrusion detection performance, they can be vulnerable to small changes in the input data. In an adversarial attack, an adversary creates slight but carefully-crafted examples to affect the ML prediction performance [4]. Fig. 1 demonstrates the impact of an adversarial attack (under varying perturbation amounts) on ML prediction performance for intrusion detection. Here, 0% perturbation refers to no adversarial attack. Higher perturbation amount implies a stronger adversarial attack. In this figure, different colors denote select ML methods, e.g., random forest (RF), hyperdimensional computing (HD), deep neural network (DNN). Under an adversarial attack, accuracy (Fig. 1a) and F_1 score (Fig. 1b) can be impacted significantly irrespective of the underlying ML model, causing up to 25× performance loss. These results motivate the need for an effective defensive mechanism against adversarial attacks for ML-based IDS.



(a) **Accuracy** (b) F_1 **Score**

Fig. 1: ML prediction performance under adversarial attack

In this work, we propose RObust Layered DEFense (*ROLDEF*) which is presented in Fig. 2. Given test data, we first use a pretrained denoising autoencoder (DAE) to predict the perturbation amount. We then mark if the data belongs to an adversarial attack depending on the perturbation amount prediction. If the input is predicted to be an attack, we perform DAE selection where the best performing DAE is chosen. The selected DAE eliminates the perturbation and passes the data to the pretrained ML algorithm. Comprehensive experiments on a realistic IIoT intrusion detection dataset [7] show that *ROLDEF* provides both robust and effective defense for various ML methods under different adversarial attacks. It can improve average model prediction performance by 114% with respect to no defense. *ROLDEF* is also consistently better than the state-of-the-art (SoA) adversarial training defense [8] with 50.1% average prediction performance improvement. While improving adversarial robustness, our defense requires additional 0.09 milliseconds per sample (on average) compared to the SoA defense. Most importantly, *ROLDEF* can be deployed for any ML-based IDS and could effectively protect the learning system against adversarial attacks.

979-8-3503-4859-0/DATE24/© 2024 EDAA

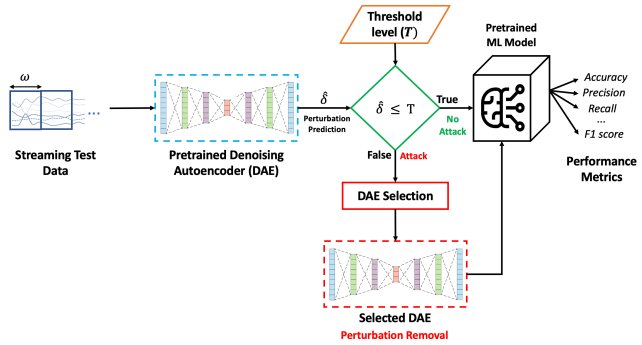


Fig. 2: Proposed defense framework (*ROLDEF*)

II. Related Work

A. IIoT Security

IIoT is an adaptation of traditional IoT for industrial environments enabling full automation, remote monitoring, and predictive maintenance [9]. Due to inadequate standardization and the lack of required skills to implement them, IIoT has become a target for different cyber attacks, e.g., denial of service, eavesdropping, man-in-the-middle, spoofing, and side channel [10]. An adversary can gain access to an entire IIoT system by exploiting its vulnerable assets such as operating systems, application software, industrial communication protocols, and smart devices [11]. There are advanced security solutions in traditional IT systems, yet these cannot be directly used in IIoT systems due to IIoT's limited power, constrained functionality, and lightweight network protocols. ML-based IDS is one possible security solution that trains ML models by using historical network data to detect attacks and anomalies [5]. There are different models proposed in the literature, e.g., logistic regression, random forest, deep neural networks [6].

B. Adversarial Attacks and Defenses

Adversarial attacks craft perturbed input data to fool ML models [4]. These attacks can significantly impact ML-based IDS decisions where benign data can be classified as an attack or vice versa. To effectively generate those instances, attacker can use white-box or black-box attacks. While white-box exploits complete knowledge of an ML model, i.e., model parameters and architecture, black-box refines adversarial input based on an output generated from the model. Adversarial defenses aim to protect ML models against adversarial attacks under three main groups [12]: (i) input defense, (ii) adversarial attack detection, and (iii) model defense. Input defense preprocesses input data to remove any adversarial component, e.g., data compression, data coding, data decomposition. Adversarial attack detection aims to distinguish adversarial attack data from clean ones before model training and inference, e.g., data feature analysis, ML-based detection. Model defense strengthens the ML model against adversarial attacks via further modifications, e.g., gradient masking, defensive distillation, adversarial training.

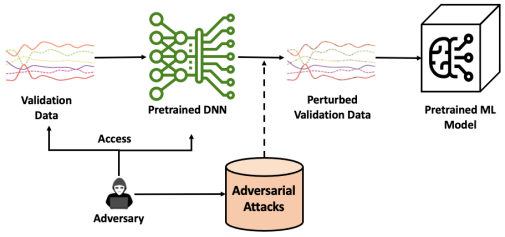


Fig. 3: Adversarial Transfer Attack Framework

C. Denoising Autoencoder (DAE) Defense

Autoencoder (AE) is an unsupervised learning setting which consists of encoder, code, and decoder. Encoder performs input compression and generates the code, and the decoder reconstructs the input from the code [13]. The AE goal is to get an output identical with the input. However, AE carries a risk of learning identity function where input is directly copied to the output without any learning performed. Denoising AE (DAE) solves this problem through noise injection. DAE first adds some random noise to the input data and reconstructs the clean data. Due to this property, DAE is capable of removing adversarial noise before target model prediction. Hence, DAE can be used as a defense against adversarial attacks where perturbed data is pre-processed before model prediction. There are several AE-based defenses in the literature: MagNet [14], deep denoising sparse autoencoder (DDSA) [15], adversarial noise removing network (ARN) [16], and adversarial purification denoising autoencoder (APuDAE) [17]. All these defenses are applied to images, and do not apply to the time series data.

III. *ROLDEF* Framework

Fig. 2 illustrates our RObust Layered DEFense framework *ROLDEF*. Our defense is based on denoising autoencoder (DAE) to remove adversarial perturbation before ML model inference. We use DAE since it can purify noisy samples by reconstructing the output that is like the clean input [17]. Given test data with a window size ω, the first step is to determine the perturbation prediction amount $\hat{\delta}$. Perturbation prediction $\hat{\delta}$ is calculated by finding the average difference between DAE input and reconstructed output. We then compare $\hat{\delta}$ with a predefined threshold level T to mark it as clean or attack data. If an attack is detected ($\hat{\delta} > T$), we select the best performing DAE (against adversarial attacks) among a set of pretrained DAEs. Finally, data is purified and given to the ML model to obtain the prediction metrics, e.g., F_1 score, accuracy. We use 6 ML methods: decision tree (DT), random forest (RF), logistic regression (LR), naive bayes (NB), hyperdimensional computing (HD) [18], and deep neural network (DNN). However, *ROLDEF* can work with any ML-based IDS system, making it a generalizable defense solution.

A. Black-box Transfer Attack Framework

Fig. 3 presents our black-box transfer attack framework. Black-box attacks represent a more realistic attack scenario where the attacker can be an outsider, with limited knowledge

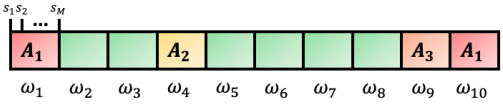


Fig. 4: Adversarial attack simulation

about the internal system [19]. Adversary first needs to create a surrogate (substitute) model and access the validation data to create adversarial examples. Attackers usually exploit network vulnerabilities to access data in IIoT systems. We select 4 different gradient-based attacks: fast gradient sign method (FGSM) [20], randomized fast gradient sign method (RFGSM) [21], projected gradient descent (PGD) [8], and momentum iterative method (MIM) [22]. We select these attacks since they are the most common attacks in time-series classification [23]. We use a deep neural network (DNN) as the surrogate model due to its superior prediction performance [7]. We train it using the training data. The attacker then uses the validation data to create attacks and send them to the target ML models. The selected target ML models are DT, RF, LR, NB, HD, and DNN. We use a separate target DNN model to make the attack black-box, i.e., the surrogate and target ML models are different. These methods are selected due to their prevalence in the IDS domain [24]. All these methods are previously trained using our training data. Given perturbed validation data, we store the prediction performance for each attack type and ML model to keep track of the individual attack effectiveness. Based on these values, we select the most effective DAE adversarial noise type as provided in Section III-E.

B. Denoising Autoencoder (DAE) Training

We perform two main modifications to the traditional DAE training: (i) injecting adversarial noise and (ii) including clean data. For the former, we use the adversarial attack data instead of some random noise such as Gaussian noise. To create the adversarial attack data, we leverage the black-box attack framework in Fig. 3 and input it to the DAE. For the latter change, we also incorporate some clean data into the training to increase DAE robustness in case there is no adversarial attack. Our experiments showed that both changes contributed to the DAE adversarial robustness positively since DAE is able to learn the reconstruction of adversarial examples with some clean data. This means that when trained DAE receives some attack and clean data, it reconstructs the data more accurately. Overall, we train four different DAEs corresponding to different adversarial attacks. We give equal weights to the clean and attack data in training, i.e., 50% clean, 50% attack data due to more accurate predictions under adversarial attacks among different ratios. As the output of DAE training, we obtain four trained DAEs which are going to be used in perturbation prediction and perturbation removal.

C. Variation in Adversarial Attacks

Adversarial attacks can change in time where an adversary can try different attacks to deceive the deployed ML method [25]. To reflect this situation, we receive the test data in

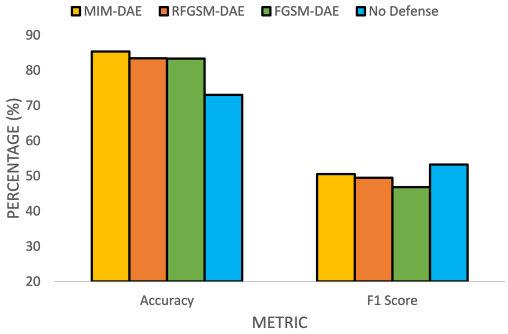


Fig. 5: DAE performance under varying adversarial attacks ($\xi = 20$, $ADR = 50\%$)

windows, with size ω. We shift each window by ω at each iteration and receive the new set of data continuously (or until all test data is consumed). Here, we assume that data is processed in windows, i.e., data is fed into the target ML model after all data from a window is received. Fig. 4 simulates an adversarial attack scenario where clean (green square) or attack data (red, yellow, and orange squares) is received during each time window, e.g., ω_1, ω_2. Each time window consists of M number of samples, i.e., $s_1, s_2, \ldots, s_M$. We observe that different attacks (denoted by distinct colors) might arrive at random times with random frequencies. Based on this observation, we consider three possible adversarial attack scenarios: (i) adversary always conducts an attack, i.e., all samples are attack, (ii) adversary never conducts an attack, i.e., all samples are clean, and (iii) adversary conducts an attack at some random time with unknown frequencies, i.e., some samples are attack/clean. Among these three scenarios, the last one represents the most realistic and stealthy attack setting since the adversary would not want to be detected by a defense mechanism. Let ξ denote the selected number of windows in a given test data. So, in a single window ω, we have N/ξ samples where N is the total number of test samples. To denote the number of times adversarial attack is conducted in total, we introduce attack data ratio (ADR) which is the number of attack windows (ξ_{attack}) divided by the total number of windows ($\xi = \xi_{attack} + \xi_{clean}$):

$$\text{Attack data ratio } (ADR) = \frac{\xi_{attack}}{\xi_{attack} + \xi_{clean}} \tag{1}$$

We select different ADRs to show the validity of the proposed defense, covering all three adversarial attack scenarios.

D. Layered Defense Motivation

Fig. 5 presents the pretrained DAEs' defense performance against varying adversarial attacks where ξ is 20, and ADR is 50%. This figure uses HD as the target ML model. Here, a random adversarial attack is selected for the attack samples and the result represents average values over all test windows. While x-axis is the prediction metric, y-axis has the corresponding values in percentage. Different colors represent DAEs trained with different adversarial attacks (e.g., *MIM-DAE* denotes DAE with MIM noise injection) and *No Defense*

is indicated by light blue color. We observe that *No Defense* outperforms DAE defenses based on F_1 score. This is due to the DAE performance under clean data. Although DAE performs well under adversarial attacks, its performance gets worse with clean data where traditional training is favorable. This means that defense should be adaptive based on the received data, i.e., clean or attack. To solve this problem, we devise a layered defense mechanism based on DAE perturbation prediction where we determine if an attack is conducted and remove the perturbation if there is an adversarial attack.

E. Perturbation Prediction

Perturbation prediction calculates the amount of perturbation in a given data. Based on the perturbation prediction $\hat{\delta}$, we categorize the data as clean or attack. Given input data $\ddot{x}$, DAE reconstructs it to obtain x'. We calculate the perturbation amount $\hat{\delta}$ by finding the absolute difference between $\ddot{x}$ and x' averaged over all samples:

$$\hat{\delta} = \frac{\sum_{i=1}^{M} abs(\ddot{x}_i - x'_i)}{M} \tag{2}$$

where abs is the absolute value function and M is the number of samples in a single window. To calculate the perturbation amount, we use the DAE trained with the most effective attack which is determined based on our attack prediction performance result from Section III-A. This is selected for each target ML method. After $\hat{\delta}$ is calculated, we determine if the data is from an attack or not using a threshold level T. If $\hat{\delta} \leq T$, we have clean data; otherwise ($\hat{\delta} > T$) we have an attack data. In case of an adversarial attack detection, we utilize our pretrained DAEs to remove the adversarial perturbation, otherwise we directly use our pretrained ML models without any further modification.

F. DAE Selection and Perturbation Removal

After we predict that the given data comes from an attack, the best DAE is selected and applied as a defense mechanism. For this selection, we denoise the perturbed data via pretrained DAEs and provide it to the target model. This model then outputs the performance metrics corresponding to individual DAEs. Based on these metrics, we select the DAE that gives the best prediction performance. The selected DAE is then used as a defense. It is important to note that the best DAE can change at each time window and for each ML method.

IV. Experimental Analysis

A. Dataset Description

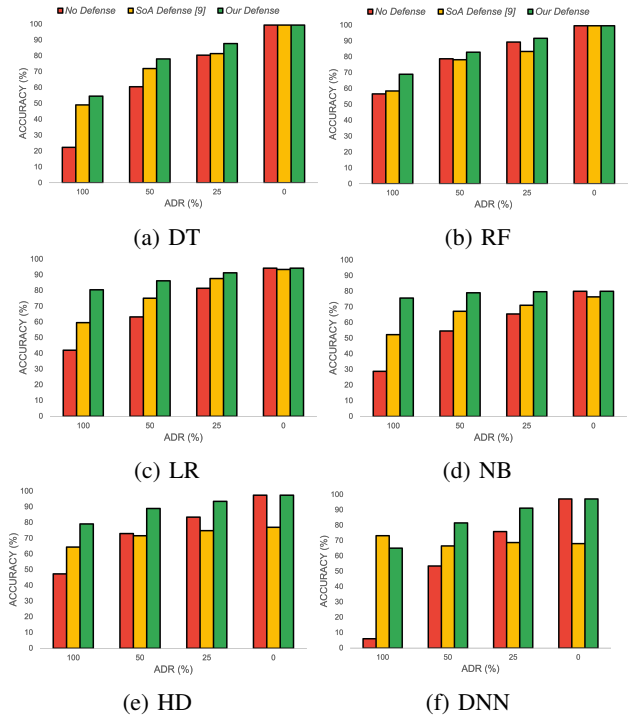
We use a realistic IIoT intrusion dataset, X-IIoTID [7]. X-IIoTID is a device and connectivity agnostic dataset which addresses the heterogeneity of IIoT network traffic and systems' activities generated from distinct connectivity protocols, devices, and communication patterns. 18 different attacks are included in the dataset: malicious insider, reverse shell, MitM attack, MQTT cloud broker-subscription, generic scanning, Modbus-Register reading, TCP relay attack, scanning vulnerabilities, fuzzing, discovering resources, brute force attack, dictionary attack, command and control, exfiltration, false data injection, fake notification, crypto-ransomware, and ransom denial of service. The input data comes from end-to-end network traffic, physical properties, host device logs, the host device's resources, and alert logs. Data collection began on December 5, 2019, ran for many hours each day, and ended on March 23, 2020 (not continuous). The overall learning goal is to map the input data to the attack labels.

B. Experimental Setup

We run all experiments on a PC with 16 GB RAM and an 8-core 2.3 GHz Intel Core i9 processor. For our surrogate model, we select a DNN with 2 hidden layers with 30, and 20 units. To train this DNN, we use *SGD* optimizer with learning rate 0.01, *ReLU* activation function, and batch size of 32. For our traditional ML target models, e.g., DT, RF, we perform a grid search to find their optimal hyper-parameters. For HD, we set HV dimension to 1000, used random projection encoding, and set learning rate to 2 after detailed experiments. Our target DNN consists of 3 hidden layers with 50, 30, and 20 units. The selected defense DAE structure contains an encoder with 32, 16, and 8 units and a decoder with 16, 32 and 58 units. DAE is trained using *Adam* optimizer, with mean squared error loss. For DAE noise injection, we add noise generated from {FGSM, RFGSM, PGD, MIM} attack set with 0.5 as the perturbation amount. We also included 50% clean and 50% attack data in training which gives the best prediction performance under various adversarial attacks. We set the number of windows (ξ) to 20. This selection brings the largest prediction improvement over no defense. We report average metrics over all windows. For varying adversarial attacks, we select a random attack from the {FGSM, RFGSM, PGD, MIM} set with the random perturbation from the interval $[0.1, 0.9]$. For ADR, we experimented with the ratios (%) from $\{0, 25, 50, 100\}$ to consider different adversarial attack scenarios. We also set the threshold level T to 0.01 to differentiate attack samples from clean ones after detailed experiments. To measure the defense performance, we select 2 metrics: accuracy and F_1 score. Since we have an unbalanced dataset (number of attack samples are relatively smaller than clean ones), F_1 score provides a more meaningful insight.

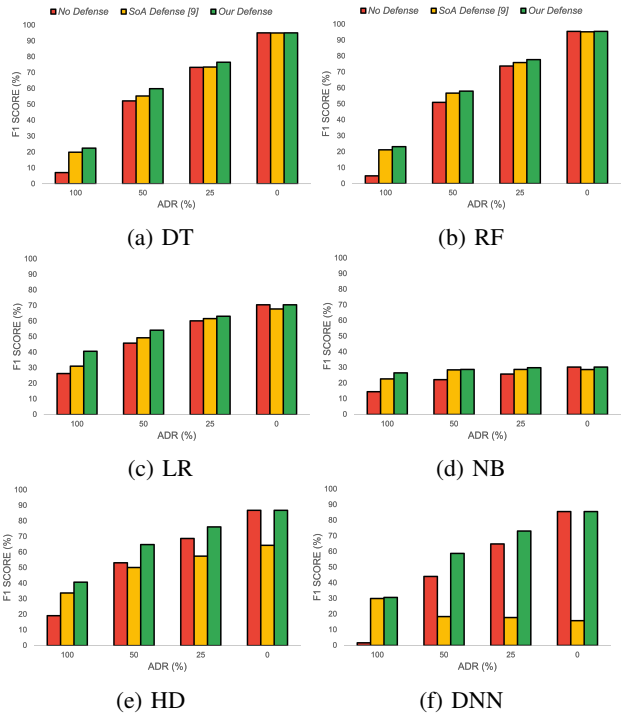
C. State-of-the-art Defense

Adversarial training (AT) [8]: AT is widely accepted as the most effective defense method against adversarial attacks [26]. We selected this as the state-of-the-art defense since it can be applied to any ML model. It incorporates adversarial attack data into the training process. For the DNN, we retrain the model via adversarial attack samples where network weights are updated. For traditional ML methods, we retrain the models by including adversarial attack data. For HD, class hyper-vectors are updated inspired by the method presented by Ma et al. [27]. For each ML method's adversarial training, we use the best adversarial attack to obtain the most effective defense performance. Hence, we report the most effective adversarial training defense in our experimental results.



(a) DT (b) RF (c) LR (d) NB (e) HD (f) DNN

Fig. 6: Accuracy Comparison Among *No Defense*, *SoA Defense (AT)* [8], and Our Defense (*ROLDEF*)



(a) DT (b) RF (c) LR (d) NB (e) HD (f) DNN

Fig. 7: F_1 Score Comparison Among *No Defense*, *SoA Defense (AT)* [8], and Our Defense (*ROLDEF*)

Defensive Distillation (DD) [28]: DD is another well-known defense approach against adversarial attacks when adversarial data is not allowed during training. This approach uses two DNNs: initial and distilled network. Initial DNN probability vector predictions are given to the distilled network and training is performed with the original labels. Trained distilled network is used as the inference model. We make a comparison with DD when the target model is DNN.

D. Experimental Results

ROLDEF Performance: Fig. 6 and Fig. 7 compare *ROLDEF* (green color) against the *SoA defense* [8] (yellow color), and *No Defense* (red color) in terms of accuracy and F_1 score respectively. Each sub-figure has varying attack data ratio (ADR) on the x-axis and y-axis is accuracy or F_1 score. In terms of accuracy (Fig. 6), our defense consistently provides the best prediction performance across various target models and ADR values. Table I presents our method's accuracy improvement over *No Defense* and *SoA defense*. Compared to *No Defense*, *ROLDEF* brings up to 965.1% (259.4% on average) accuracy improvement when the target model is DNN. With respect to *SoA defense*, *ROLDEF* provides up to 44.6% (24.7% on average) accuracy improvement. Compared to *No Defense* and *SoA defense*, we can obtain 71.7% and 15.9% accuracy improvement over all target models respectively.

In terms of F_1 scores (Fig. 7), we can observe the superiority of our defense. *ROLDEF* is consistently the best approach when $ADR > 0$. Table II shows our method's F_1 score improvement over *No Defense* and *SoA defense*. Compared to *No Defense*, *ROLDEF* brings up to 1706.3% (438.1% on average) F_1 score improvement. With respect to *SoA defense*, *ROLDEF* provides up to 440.6% (242.9% on average) F_1 score improvement. We can obtain 114% and 50.1% F_1 score improvement over *No Defense* and *SoA defense* when all target models are considered. Overall, we can note that our defense consistently improves the prediction performance irrespective of the underlying target ML model.

TABLE I: *ROLDEF* Accuracy Improvement (%)

	No Defense		*SoA Defense [8]*	
Target Model	**Maximum**	**Average**	**Maximum**	**Average**
DT	144.6	45.7	11.3	6.9
RF	21.8	7.4	18.0	8.5
LR	91.4	35.0	35.1	13.7
NB	162.4	57.2	**44.6**	19.8
HD	67.0	25.2	26.6	**24.7**
DNN	**965.1**	**259.4**	42.7	21.7

TABLE II: *ROLDEF* F_1 Score Improvement (%)

	No Defense		*SoA Defense [8]*	
Target Model	**Maximum**	**Average**	**Maximum**	**Average**
DT	219.1	59.6	13.1	6.4
RF	375.7	98.8	9.1	3.5
LR	54.4	19.4	30.9	11.8
NB	82.6	31.9	17.0	6.8
HD	112.1	36.2	34.9	29.3
DNN	**1706.3**	**438.1**	**440.6**	**242.9**

Defense Comparison under DNN Target Model: Based on Table II, the largest improvement is observed when the target model is DNN. This result can be attributed to the

TABLE III: Defense Comparison under DNN Target Model

	Accuracy			F1 Score		
ADR (%) / Defense	***AT** [8]*	***DD** [28]*	***ROLDEF***	***AT** [8]*	***DD** [28]*	***ROLDEF***
100	73.3	26.8	65.1	30.1	9.4	30.7
50	66.6	60.6	81.6	18.5	46.1	58.9
25	68.7	78.9	91.2	17.9	64.9	73.2
0	68.1	96.8	97.2	15.8	83.5	85.6
Average	**69.2**	**65.8**	**83.8**	**20.6**	**50.9**	**62.1**

TABLE IV: *ROLDEF* Overhead Analysis

Target Model	DT	RF	LR	NB	HD	DNN
Additional Overhead (ms)	0.08	0.14	0.09	0.09	0.11	0.06

similarity between surrogate and target models. Since the adversarial attack data is crafted via a DNN surrogate, it can fool the target DNN the most although they have different network structures. To further demonstrate the effectiveness of our defense under DNN target model, we compare *ROLDEF* with defensive distillation (DD) [28]. Table III presents the results of this comparison. We observe that our defense provides the best prediction performance under all ADR values, solidifying the case for the superiority of our defense.

Overhead Analysis: *ROLDEF* overhead is the sum of perturbation prediction and DAE selection. Table IV shows our method's additional per sample overhead over the selected ADR values compared to the *SoA defense*. When average execution time is calculated over all target ML methods, running our defense requires an additional 0.09 milliseconds per sample compared to the *SoA defense*. For our data set, a sample refers to the time interval between the last and the first packet seen in a network traffic flow [7], where the median traffic flow duration is 4.98 ms. Compared to the median traffic flow duration, our method's overhead is around 1.8%.

V. Conclusion

IIoT security is challenging owing to its large attack surface, and increased inter-connectivity. Intrusion Detection Systems (IDSs) dynamically monitor the behavior of an IIoT system to detect malicious activity. ML-based IDS solution is popular owing to its great prediction performance. However, ML methods are sensitive to adversarial attacks, impacting their prediction performance significantly. These attacks can happen with unknown perturbations and frequencies. Hence, defense should be adaptive against attacks. In this paper, we proposed a robust layered defense against adversarial attacks. Our DAE-based defense first detects if sample comes from an attack. If attack is detected, adversarial noise is eliminated using the best DAE. Our defense improved average model prediction performance by 114% and 50% with respect to no defense and the state-of-the-art adversarial training defense.

Acknowledgement

This work has been funded in part by NSF, with award numbers #1911095, #2003277, #2003279, #2100237, #2112167, #2112665, and in part by PRISM and CoCoSys, centers in JUMP 2.0, an SRC program sponsored by DARPA.

References

[1] E. Sisinni *et al.*, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE transactions on industrial informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[2] P. Daugherty and B. Berthon, "Winning with the industrial internet of things: How to accelerate the journey to productivity and growth," *Dublín: Accenture*, 2015.

[3] K. Tange *et al.*, "A systematic survey of industrial internet of things security: Requirements and fog computing opportunities," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2489–2520, 2020.

[4] O. Gungor *et al.*, "Stewart: Stacking ensemble for white-box adversarial attacks towards more resilient data-driven predictive maintenance," *Computers in Industry*, vol. 140, p. 103660, 2022.

[5] E. Anthi *et al.*, "Adversarial attacks on machine learning cybersecurity defences in industrial control systems," *Journal of Information Security and Applications*, vol. 58, p. 102717, 2021.

[6] H. Liu and B. Lang, "Ml and dl methods for intrusion detection systems," *applied sciences*, vol. 9, no. 20, p. 4396, 2019.

[7] M. Al-Hawawreh *et al.*, "X-iiotid: A connectivity-agnostic and device-agnostic intrusion data set for industrial internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3962–3977, 2021.

[8] A. Madry *et al.*, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[9] O. Gungor, T. S. Rosing, and B. Aksanli, "Dowell: diversity-induced optimally weighted ensemble learner for predictive maintenance of industrial internet of things devices," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 3125–3134, 2021.

[10] M. Lezzi *et al.*, "Cybersecurity for industry 4.0 in the current literature," *Computers in Industry*, vol. 103, pp. 97–110, 2018.

[11] D. Wu *et al.*, "Cybersecurity for digital manufacturing," *Journal of manufacturing systems*, vol. 48, pp. 3–12, 2018.

[12] J. Li, Y. Liu, T. Chen, Z. Xiao, Z. Li, and J. Wang, "Adversarial attacks and defenses on cyber–physical systems: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5103–5115, 2020.

[13] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[14] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pp. 135–147, 2017.

[15] Y. Bakhti, S. A. Fezza, W. Hamidouche, and O. Déforges, "Ddsa: A defense against adversarial attacks using deep denoising sparse autoencoder," *IEEE Access*, vol. 7, pp. 160397–160407, 2019.

[16] D. Zhou *et al.*, "Towards defending against adversarial examples via attack-invariant features," in *International Conference on Machine Learning*, pp. 12835–12845, PMLR, 2021.

[17] D. Kalaria *et al.*, "Towards adversarial purification using denoising autoencoders," *arXiv preprint arXiv:2208.13838*, 2022.

[18] L. Ge and K. K. Parhi, "Classification using hyperdimensional computing: A review," *IEEE Circuits and Systems Magazine*, vol. 20, no. 2, pp. 30–47, 2020.

[19] S. Bhambri *et al.*, "A survey of black-box adversarial attacks on computer vision models," *arXiv preprint arXiv:1912.01667*, 2019.

[20] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[21] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," *arXiv preprint arXiv:2001.03994*, 2020.

[22] Y. Dong *et al.*, "Boosting adversarial attacks with momentum," in *CVPR*, pp. 9185–9193, 2018.

[23] H. I. Fawaz *et al.*, "Adversarial attacks on deep neural networks for time series classification," in *IJCNN*, pp. 1–8, IEEE, 2019.

[24] P. Mishra *et al.*, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE communications surveys & tutorials*, vol. 21, no. 1, pp. 686–728, 2018.

[25] Y. Gong, B. Li, C. Poellabauer, and Y. Shi, "Real-time adversarial attacks," *arXiv preprint arXiv:1905.13399*, 2019.

[26] T. Bai *et al.*, "Recent advances in adversarial training for adversarial robustness," *arXiv preprint arXiv:2102.01356*, 2021.

[27] D. Ma *et al.*, "Hdtest: Differential fuzz testing of brain-inspired hyperdimensional computing," in *2021 DAC*, pp. 391–396, IEEE, 2021.

[28] N. Papernot *et al.*, "Distillation as a defense to adversarial perturbations against deep neural networks," in *IEEE symposium on security and privacy (SP)*, pp. 582–597, 2016.