Robust Data-Driven Detection of Electricity Theft Adversarial Evasion Attacks in Smart Grids

Abdulrahman Takiddin, *Graduate Student Member, IEEE*, Muhammad Ismail, *Senior Member, IEEE*, and Erchin Serpedin, *Fellow, IEEE*

***Abstract*—Existing machine learning-based detectors of electricity theft cyberattacks are trained to detect only simple traditional types of cyberattacks while neglecting complex ones like evasion attacks. This paper analyzes the robustness of electricity theft detectors against evasion attacks. Such attacks decrease the reported electricity reading values and fool the electricity theft detectors by injecting adversarial samples. We propose strong evasion attacks that fool the benchmark detectors by iteratively generating adversarial samples based on an electricity reading and its neighboring readings. We study the impact of evasion attacks using white, gray, and black-box settings based on the attacker's knowledge about the detector's parameters or datasets. Our investigations revealed that the performance degradation of benchmark detectors is up to 35.8%, 26.9%, and 22.2% in white, gray, and black-box settings, respectively. To enhance the detection robustness, we propose an ensemble learning-based anomaly detector trained only on benign data to detect unseen attacks (traditional and evasion) by sequentially combining an attentive autoencoder, convolutional-recurrent, and feed forward neural networks. The proposed model offers a stable detection performance where the average degradation is only 0.7 − 3%, 0.9−2.1%, and 0.4−1.7% in white, gray, and black-box settings, respectively, with maximum adversarial sample injection levels.**

***Index Terms*—Electricity theft, machine learning, adversarial samples, evasion attacks, smart grids, robust detection.**

Nomenclature

$\boldsymbol{x}$	Input electricity reading
$\boldsymbol{y}$	True original label
$\hat{\varepsilon}$	Maximum perturbation magnitude
Λ	Standard Euclidean (root-mean-square) distance
ϕ	Model parameters
ψ	Detection threshold value
ε	Perturbation magnitude
$E_c(d, t)$	Consumption of customer c during day d and time t
$f(\cdot)$	Traditional cyberattack function
k	Number of nearest neighbors
R_c	Reported consumption
R_c^{adv}	Adversarial sample
x_{A}	Reconstructed output

I. Introduction

Electricity thefts take place when malicious customers conduct malevolent activities to reduce their electricity bills. Such actions impact the economy and lead to \$6 billion of annual losses in the United States [2] as well as overloading the power grid [3]. To counter such destructive actions, smart meters, which are part of advanced metering infrastructures, are installed at customers' premises to routinely monitor energy consumption. However, such meters only overcome the physical attacks (e.g., line hooking and meter tampering). Since smart meters are embedded systems running on software programs, they are vulnerable to cyberattacks launched by malicious customers, who connect to their smart meters to manipulate the electricity consumption reports. Detecting such attacks is challenging due to their variety and complexity.

A. Related Work and Limitations

In the literature, data-driven techniques including statistical methods as well as machine learning-based techniques (i.e., shallow and deep models) provided promising results in detecting traditional electricity theft cyberattacks.

1) Statistical Methods: A detector based on clustering and local outlier factor provided an area under the curve (AUC) score of 81% [4]. A principal component analysis-based detector provided a detection rate (DR) of up to 90% [5]. A Kullback-Leibler divergence-based detector identified attacks for 94% of customers [6]. A privacy-preserving energy theft detector offered a success rate of 95% [7].

2) Shallow Detectors: Shallow detectors use shallow machine learning-based models to identify malicious electricity readings. Detectors trained on labeled benign and malicious readings reported different performance levels. Detectors based on Naïve Bayes [8] and 2-class support vector machine (SVM) [2] offered DRs of 80% and 94%, respectively. Different shallow boosting methods including adaptive boosting with SVM [9], extreme gradient boosting [10], and

Manuscript received 1 November 2021; revised 21 March 2022 and 28 June 2022; accepted 22 July 2022. Date of publication 26 July 2022; date of current version 22 December 2022. This publication was made possible by NPRP12S-0221-190127 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors. Part of this work was presented in the 2021 IEEE International Conference on Communications (ICC) [DOI: 10.1109/ICC42927.2021.9500822]. Paper no. TSG-01751-2021. *(Corresponding author: Abdulrahman Takiddin.)*

Abdulrahman Takiddin and Erchin Serpedin are with the Electrical and Computer Engineering Department, Texas A&M University, College Station, TX 77843 USA (e-mail: abdulrahman.takiddin@tamu.edu; eserpedin@tamu.edu).

Muhammad Ismail is with the Department of Computer Science, Tennessee Tech University, Cookeville, TN 38505 USA (e-mail: mismail@tntech.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSG.2022.3193989.

Digital Object Identifier 10.1109/TSG.2022.3193989

multiple boosting techniques with feature engineering [11] provided 80% in accuracy, 97% in precision, and 97% in DR, respectively. A random forests approach provided an F1-score of 81% [12]. Employing decision trees reported 87% [13] and 92% [14] accuracy scores. A detector based on an autoregressive integrated moving average (ARIMA) [15] trained only using benign readings, provided DR of 77%.

3) Deep Detectors: Deep detectors use deep neural networks to identify malicious electricity readings. Hybrid detectors based on recurrent neural networks (RNNs) and convolutional neural networks (CNNs) offered 89% in accuracy [16], [17]. Detectors based on feed forward [18], RNNs [19], deep belief networks [20], and vector embeddings [21] presented 92% - 95% in DR. Another hybrid model combining GoogLeNet and an RNN presented an AUC score of 96% [22]. A stacked detector combining shallow models with a convolutional network offered an F1-score of 95% [23]. Autoencoder-based detectors trained on benign readings provided DRs of 81% - 94% [24], [25], [26].

The detection performance of existing statistical methods and shallow detectors is limited since they do not fully capture the temporal aspect and complex patterns within the electricity readings. While deep detectors might capture such aspects, they are still vulnerable to complex cyberattacks. Hence, the common limitation among existing detectors is that the performance is reported when the detectors are only tested against simple traditional electricity theft cyberattacks such as partial reduction and selective bypass attacks. Existing detectors have not been tested against more sophisticated types of cyberattacks that are harder to detect such as adversarial attacks. Adversarial attacks could take place during the training stage (i.e., data poisoning attacks), a topic studied in [27] or during the testing stage (i.e., evasion attacks), which is the focus of this paper. Evasion attacks are sophisticated attacks since they are designed in a way to fool the detector, go undetected, and hence, deteriorate the detection performance.

Several studies investigated the impact of adversarial samples on machine learning-based detectors deployed in the power systems domain applications such as network security situation awareness [28], state estimation [29], or household energy forecasting [30]. These studies reached the conclusion that such detectors are indeed vulnerable to evasion attacks. However, to the best of our knowledge, there is no effective solution yet to detect such attacks and enhance the robustness of electricity theft detectors against unseen evasion attacks. Also, the influence of electricity theft evasion attacks on smart meters using different attack levels and settings has not been investigated in the literature.

B. Contributions

To counter the limitations of existing detection schemes, we provide a comprehensive study on the impact of evasion attacks on benchmark detectors in multiple settings and levels. We propose strong evasion attacks that can better fool the detectors compared to benchmark evasion attacks. In addition, we propose a robust detector that can detect both unseen simple (traditional) and complex (evasion) cyberattacks. The contributions of our work are next summarized:

- To quantify the influence of evasion attacks, we investigate the performance of several stand-alone benchmark detectors with shallow architectures like ARIMA, 1-class SVM, and 2-class SVM as well as deep architectures such as deep feed forward, long-short-term-memory (LSTM), and attentive autoencoder (AAE). The detectors are also examined when equipped with state-of-the-art adversarial defense mechanisms including adversarial training [31], certified defenses [32], MagNet [33], and generative adversarial nets (GANs) [34].
- The conducted experiments include multiple injection levels of evasion attacks using various assumptions regarding the attacker's knowledge about the energy consumption dataset and the electricity theft detector's parameters. These assumptions include white, gray, and black-box settings in which the attacker has full, limited, and no knowledge, respectively.
- We propose two strong evasion attacks, namely, the nearest neighbor perturbation (NNP) attack that depends on the average perturbation value, and the nearest neighbor distance (NND) attack that depends on the average Euclidean distance of a customer reading and its surrounding readings within the same day. The proposed attacks are strong since they fool the detector with small unnoticed perturbation values iteratively that create similar readings with similar patterns as the original ones. We test these attacks using the white, gray, and black-box settings by generating and injecting adversarial samples into the test set with different levels of attack. When 50% of the test set contains adversarial samples, the performance of the benchmark detectors decreases by $33.4 - 35.8\%$, $24 - 26.9\%$, and $19.2 - 22.2\%$ in the white, gray, and black-box settings, respectively, with maximum injection level of the proposed evasion attacks.
- We compare the impact of the proposed evasion attacks to benchmark evasion attack types, namely, fast gradient sign method (FGSM) [31], basic iterative method (BIM) [35], AutoAttack (AA) [36], and Carlini & Wagner method (C&W) [37], which fool the detector by subtracting constant bounded values from the energy readings. The detection performance of benchmark detectors degrades by $24.7{-}29.2\%$, $15.2{-}19.8\%$, and $10.4{-}15.1\%$ in the white, gray, and black-box settings, respectively. The proposed attacks are stronger since they degrade the performance further by $6.5 - 8.7\%$, $7.1 - 8.8\%$, and $7 - 8.7\%$ in the white, gray, and black-box settings, respectively, with maximum injection levels. The addition of adversarial defense mechanisms offers a detection improvement of $1.1 - 3.1\%$ to benchmark detectors.
- We design a robust electricity theft anomaly detector that provides a more stable performance against high levels of evasion attacks compared to benchmark detectors. This is achieved by fusing an AAE, convolutional-recurrent, and feed forward model using sequential ensemble learning. The proposed detector is trained only on benign data, and is able to detect unseen attacks (traditional and evasion)

based on the deviation from the learned benign patterns without the need of generating and training on malicious samples. The robustness of the proposed detector is verified using 15 unseen cyberattacks where its performance deteriorates only by $0.7-3\%$, $0.9-2.1\%$, and $0.4-1.7\%$ in the white, gray, and black-box settings, respectively, with maximum injection level of strong evasion attacks.

The remaining sections of this paper are organized as follows. Section II describes the dataset preparation. Section III introduces the proposed and benchmark evasion attacks. Section IV reports the effect of evasion attacks on benchmark detectors. Section V presents the design of the proposed robust detector and reports the influence of evasion attacks on it. Section VI concludes the paper.

II. Dataset Preparation

This section introduces the target and substitute datasets. We assume that the target dataset is used by the utility company/operator to train and test the electricity theft detector. The target dataset is also used by attackers in the white-box setting to develop evasion attacks. The substitute dataset is used by attackers in the gray and black-box settings to create evasion attacks, which will then be injected into the target dataset. Both datasets capture various customers' electricity usage patterns of numerous appliances over weekdays, weekends, vacations, and all seasons.

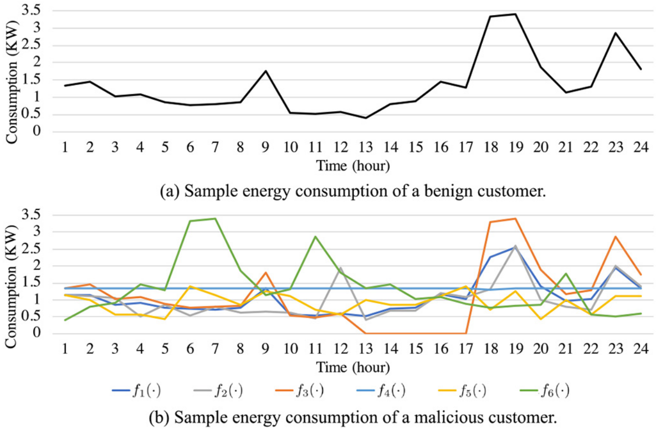
A. Target Dataset

The target dataset consists of benign and malicious datasets. The target benign energy consumption dataset is adopted from the public Irish Smart Energy Trial (ISET) dataset [38]. The target malicious data contains malicious readings generated using six traditional cyberattack functions [2].

1) Target Benign Dataset: We utilize the ISET dataset in order to train and test the detectors. The Sustainable Energy Authority of Ireland [38] published this dataset, which contains $25,000$ electricity reports in kWh per customer obtained from $3,000$ smart meters. The readings are reported half-hourly for 18 months. Consider matrix $\boldsymbol{E}_c$ whose entry $E_c(d,t)$ denotes the value of electricity consumption for customer c during day d and time period t. For benign customers, the reported energy $R_c(d,t)$ is identical to the actual consumed energy $E_c(d,t)$. Sample benign electricity readings are plotted in Figure 1a.

2) Target Malicious Dataset: For a malicious customer, $R_c(d,t)$ is different than $E_c(d,t)$. Creating the malicious dataset is carried out by utilizing the approach of false data injection [2], where six traditional cyberattack functions $f(\cdot)$ are adopted to generate $R_c(d,t)$. The adopted attacks mimic a wide range of realistic malicious behaviors that are based on partial reduction ($f_1(\cdot)$ and $f_2(\cdot)$), selective bypass ($f_3(\cdot)$), and price based load control ($f_4(\cdot)$, $f_5(\cdot)$, and $f_6(\cdot)$).

- $f_1(E_c(d,t)) = \alpha E_c(d,t)$ reduces $E_c(d,t)$ by a constant fraction ($\alpha < 1$).
- $f_2(E_c(d,t)) = \beta(d,t)E_c(d,t)$ decreases $E_c(d,t)$ by a dynamic fraction ($\beta(d,t) < 1$).



(a) Sample energy consumption of a benign customer.

(b) Sample energy consumption of a malicious customer.

Fig. 1. Target dataset sample consumption readings.

- $f_3(E_c(d,t))$ changes $E_c(d,t)$ to zero during $[t_i(d), t_f(d)]$ and otherwise it reports the real consumption, i.e.,

$$f_3(E_c(d,t)) = \begin{cases} 0 & \forall t \in \left[t_i(d), t_f(d)\right] \\ E_c(d,t) & \forall t \notin \left[t_i(d), t_f(d)\right]. \end{cases}$$

- $f_4(E_c(d,t)) = \mathbb{E}[E_c(d)]$ reports a fixed consumption value across the day. $\mathbb{E}[\,\cdot\,]$ depicts the averaging operator.
- $f_5(E_c(d,t)) = \beta(d,t)\mathbb{E}[E_c(d)]$ reports a dynamic fraction ($\beta(d,t) < 1$) of $\mathbb{E}[E_c(d)]$.
- $f_6(E_c(d,t)) = E_c(d, T-t+1)$ rearranges the reported values in a way that higher consumption is reported when the electricity price is low.

We apply all of the cyberattack functions $f(\cdot)$ to the electricity consumption profile matrix $\boldsymbol{E}_c$ of each customer, which results in producing six malicious matrices per customer. Each matrix row represents the consumption profile sample during the day. Each row has a label; if the sample is benign, the label is 0; otherwise, the label is 1. Sample malicious electricity readings are illustrated in Figure 1b.

3) Train and Test Data: All investigated detectors herein are generalized. Thus, to train and test the detectors, we merge all customers' data together [27]. The smart meter data is then split into train and test sets. For the detectors trained only on benign readings, we concatenate all customer data and then using a 2:1 ratio we split them into disjoint train and test sets. For testing, the malicious samples are concatenated with the benign test set, which might lead to result misinterpretation due to the presence of more malicious samples than benign ones. To balance the data, the adaptive synthetic (ADASYN) [39] sampling approach is adopted so that the minor class is over-sampled. To ensure that all customer samples present equal influence during training, normalized feature scaling is applied. As a result, the training set X_{TR} as well as the test set X_{TST} with Y_{TST} labels are scaled with zero-mean and unit-variance.

The rest of the studied models are 2-class classifiers trained and tested on benign and malicious data. All customers' samples (benign and malicious) are concatenated. To balance them, ADASYN is employed. Then, using a 2:1 ratio, we split the concatenated dataset into disjoint train and test sets. Using

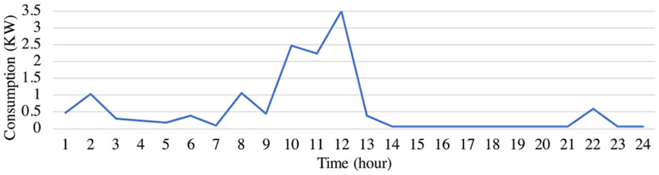


Fig. 2. Substitute dataset sample consumption readings.

normalized feature scaling, we obtain scaled X_{TR} with label Y_{TR} and X_{TST} with label Y_{TST}.

B. Substitute Dataset

In gray and black-box settings, the attacker does not have access to the target data that the utility company uses to train and test the electricity theft detector. Therefore, the attacker tends to use a substitute dataset to simulate attacks and apply the attacks to fool the implemented detector by the utility company. To mimic the behavior of the attacker when designing evasion attacks in such settings, we use the public Australian Smart-Grid-Smart-City Customer Trial (ASCT) dataset [40]. The ASCT data contains half-hourly interval meter readings of households in kWh for $13,000$ benign customers over a period of 2 years. Sample electricity readings of a customer from ASCT are illustrated in Figure 2.

C. Attack Levels

To study the impact of traditional and evasion attacks on the detectors' performance in the testing stage, we carry out attacks at five different injection levels of traditional (using the six attack functions introduced in Section II-A2) and evasion (whether adversarial samples are generated using the target or substitute dataset) attacks. The test data X_{TST} contains 50% benign samples, which is constant in all the tests. The remaining 50% is divided based on the percentage of injected adversarial samples as follows. The first case denotes 0% evasion and 100% traditional attacks. The second case denotes 25% evasion and 75% traditional attacks. The third case denotes 50% of each type. The fourth case denotes 75% evasion and 25% traditional. The last case denotes 100% evasion and 0% traditional attacks. This way, we capture various attack injection levels and report the performance accordingly.

III. Evasion Attacks

Evasion attacks refer to designing malicious electricity readings in a way that fools the detector, and thus, it falsely reports them as being benign. Adversarial samples are generated by applying evasion attack functions and injecting them into the test set [31]. We first propose two strong evasion attacks based on the k-nearest neighbors algorithm, namely, the NNP attack that depends on the average perturbation value and the NND attack that depends on the average Euclidean distance of a reading and its surrounding readings of the customer within the same day. We then compare the impact of the proposed evasion attacks to benchmark evasion attacks, namely, FGSM [31], BIM [35], AA [36], and C&W [37].

TABLE I
Attacker's Knowledge in the Evasion Attack Settings

Setting	Knowledge Level	Access	Used Detector	Used Dataset
White-box	Full knowledge	Utility's detector and dataset	Same as utility	Same as utility
Gray-box	Partial knowledge	Utility's detector only	Same as utility	Substitute
Black-box	No knowledge	No access	Substitute	Substitute

A. Settings of Evasion Attacks

In order to cover real-life scenarios, we capture all types of possibilities about the knowledge that the attacker has about the employed machine learning-based electricity theft detector and datasets by the operator [41]. Knowledge about the detector includes knowing the used detection algorithm in terms of the network architecture and parameters. Knowledge about the datasets includes gaining access to the target training and testing datasets. Table I summarizes the evasion attack settings along with the attackers' knowledge and access as well as the used detector and dataset by the attackers. Specifically, in a white-box setting, the attacker has full knowledge about the dataset and the detector's architecture and uses the gradients to generate adversarial samples [42]. In a gray-box setting, the attacker has partial knowledge [43], which includes knowledge about the used electricity theft detector only, without access to the target datasets. In a black-box setting, the attacker does not have knowledge [44] about the used datasets nor the detector. Attackers in white and gray-box settings are insiders (i.e., employees at the utility company) since they have full or partial access to the detection details. An attacker in a black-box setting is considered as an outsider (i.e., a malicious customer without any inside information). Hence, white-box attackers use the target dataset, whereas the gray and black-box attackers use the substitute dataset to create evasion attacks.

B. Proposed Evasion Attacks

We propose evasion attack functions to create adversarial samples and investigate their influence on the detectors. Such attacks use a series of dynamic perturbation values that depend on a target electricity reading $E_c(d, t)$ (i.e., a reading the attacker aims to manipulate) and k surrounding readings of the customer within same the day. The resulting perturbation is a small, yet effective, value that is subtracted from $\boldsymbol{E}_c$ to fool the detector, and hence, reduces the reported consumption value without being detected due to exhibiting similar patterns to the original reading. The main difference between the traditional cyberattacks introduced in Section II-A2 and evasion attacks lies in the detectability of the attacks.

1) NNP Attack: This attack generates adversarial samples using a perturbation value ε [31]. To obtain ε for the target electricity reading $E_c(d, t)$, NNP uses the gradient of the loss function of the model with respect to $E_c(d, t)$ as well as k surrounding readings of the customer across the same day. This is done to generate a similar reading R_c^{adv} that maximizes that loss. Finding a malicious reading that maximizes the model's loss translates into a high probability for the theft to be undetected. This is carried out in an iterative process such that

$$R_c^{\text{adv}}(d, t+1) = \text{Clip}_{E_c(d,t),k}\{R_c^{\text{adv}}(d, t) - \varepsilon \text{ sign}\Big(\nabla_{E_c(d,t)} J\Big(\phi, R_c^{\text{adv}}(d, t), \boldsymbol{y}\Big)\Big)\}, \quad (1)$$

where we apply the clip function after each time step t to guarantee that the generated and original readings present similar patterns [35]. ∇_{E_c} bespeaks the model gradient, J is the model's loss function, ϕ designates the model parameters, and $\boldsymbol{y}$ identifies the true original label. To find ε for $E_c(d, t)$, in a sample series of readings where (e.g., $k = 4$), $\mathcal{E}_c = [E_c(d, t-2), E_c(d, t-1), E_c(d, t), E_c(d-1, t+1), E_c(d-1, t+2)]$, we obtain $\bar{\mathcal{E}}_c$, which is the average value of the readings in $\mathcal{E}_c$. ε at time t is $\varepsilon = \bar{\mathcal{E}}_c \ E_c(d, t)$. This guarantees that ε changes at each reading as each reading has different surrounding readings with different average values.

2) NND Attack: Unlike the NNP attack, which uses a perturbation value to fool the detector, the NND attack fools the detector by formulating an adversarial sample for $E_c(d, t)$ based on the minimization of the Euclidean distance (root-mean-square) Λ [37]. In the NND attack, for each generated sample, Λ varies as it depends on the average of $E_c(d, t)$ and k neighboring readings. Hence, ε differs for each generated R_c^{adv}, where Λ denotes the Euclidean distance between $E_c(d, t)$ and ε such that $\Lambda(E_c(d, t), \varepsilon)$ where $\varepsilon = \bar{\mathcal{E}}_c \ E_c(d, t)$. The generation of R_c^{adv} is expressed as follows

$$R_c^{\text{adv}}(d, t) = \min_{\varepsilon} \Lambda(E_c(d, t), \varepsilon). \quad (2)$$

The goal is to find ε that minimizes $\Lambda(E_c(d, t), \varepsilon)$ with a small value of ε that would turn the reading into a benign reading without being detected.

C. Benchmark Evasion Attacks

We also test the robustness of the detectors against the following benchmark evasion attack functions.

1) FGSM Attack: This attack generates R_c^{adv} based on a constant ε to fool the detector [31] using the gradient of the loss function of the model with respect to $E_c(d, t)$ to generate similar R_c^{adv} that maximizes that loss. This is carried out based on a one-step gradient update along the direction of the gradient's sign at each time step t. This process, where sign denotes the signum function, is expressed as:

$$R_c^{\text{adv}}(d, t) = E_c(d, t) - \varepsilon \text{ sign}\big(\nabla_{E_c(d,t)} J(\phi, E_c(d, t), \boldsymbol{y})\big), \quad (3)$$

2) BIM Attack: This attack is applied over small time steps and clips the acquired time series elements after each time step t [35]. It generates adversarial samples with similar patterns to the original ones using small perturbations in an iterative manner. Generating R_c^{adv} using BIM is similar to (1), but ε is bounded by a maximum perturbation magnitude $\hat{\varepsilon} = 0.1$.

3) AutoAttack: AA is launched by combining four evasion attack functions [36] that are based on projected gradient descent (PGD) [45], fast adaptive boundary (FAB) [46], and square attack (SA) [47]. AA extends the PGD-based attacks using cross entropy (CE) and difference of logits ratio (DLR) losses to launch two step size-free attacks, namely, APGD_{CE} and APGD_{DLR} to generate R_c^{adv} [36]. The FAB-based attack generates R_c^{adv} by minimizing the norm of the ε value needed to flip the detector's decision [46]. The SA applies bounded ε randomly at each time step t [47].

4) C&W Attack: This attack fools the detector by formulating R_c^{adv} based on the minimization of Λ between $E_c(d, t)$ and $E_c(d, t) - \varepsilon$ [37]. Generating R_c^{adv} using C&W is similar to (2), but using a constant Λ depending on one reading.

5) Limitations of Benchmark Evasion Attacks: The general limitation of the benchmark evasion attacks is that the perturbation value is generated using only one electricity reading (i.e., the target reading that the attacker aims to manipulate). Also, in FGSM, the perturbation value is constant, which increases the chances of being spotted by the detector. BIM and AA apply an iterative procedure, but still uses bounded perturbation values, which may also be detectable. C&W uses the Euclidean distance to generate an adversarial sample based on only one target reading, which may also be detectable. Thus, designing stronger evasion attacks and testing detectors' robustness against them is needed. We achieve this by designing stronger evasion attacks that can better fool the detectors with small dynamic unbounded ε values that create adversarial samples with similar patterns as the original ones.

D. Evasion Parameters

Finding the ideal attack parameters is similar in the white and gray-box settings since attackers in both settings have access to the implemented detector's details. The difference is that attackers in the white-box setting use the target dataset, whereas attackers use the substitute dataset in the gray-box setting. Hence, the easiest way to fool a detector is by performing a trial-and-error approach to find the most ideal ε that fools the model. This approach works by injecting adversarial samples into the test set using initially a small ε and observing the false negative rate (FNR) provided by the detector as ε increases. The largest ε that yields the highest FNR is considered ideal since it minimizes the reported electricity consumption while fooling the detector. For this experiment, we set ε to be between 0.1 and 0.9 with 0.01 increments since points below 0.1 are negligible and points above 0.9 are easily detectable as they present dissimilar patterns as the original readings. In black-box settings, attackers tend to keep ε small to stay undetectable while fooling the detector. Black-box attackers cannot verify the highest FNR provided by the detector since it is inaccessible to them. So, ε is set to 0.5.

In the proposed attacks, setting the value of k determines ε and Λ, where k is an even number that represents the surrounding electricity readings before and after $E_c(d, t)$. Obtaining the ideal value of k in the white and gray-box setting is done using the trial-and-error approach to find the largest value of k that maximizes ε while fooling the model by starting to inject adversarial samples using small k and increasing k until the highest FNR is reached. Since we have 36 electricity readings daily, we set the lowest and highest possible values of k to 2 and 16, respectively. In black-box settings, attackers tend to set small values for k to stay undetectable while fooling the detector. Black-box attackers cannot verify the highest FNR provided by the detector since it is inaccessible to them. Hence, $k = 2$ so that the reported values have similar averages as the real ones. This way, ε stays small while fooling the detector through small dynamic values that exhibit similar patterns as the original readings.

Since attackers' knowledge in a black-box setting is restricted [43], [44], they may or may not end up using the same detection model type employed by the operator, and hence, we consider two cases. In Section IV, we study the perfect-match case, where both the attacker and operator adopt the same type of detector (e.g., both use an SVM-based detector). In this case, since the attacker does not have access to the operator's dataset or the detector's parameters, the attacker uses the substitute dataset on the same type of detector (e.g., SVM) to create the attack data. In Section V-C5, we consider the more realistic case of a detector type-mismatch, where the attacker adopts a different detector type than the operator (e.g., the operator adopts an ARIMA detector but the attacker who does not know that, assumes a feed forward detector). Here, the attacker uses the substitute dataset on the assumed detector type (e.g., feed forward) to create the attack data.

IV. Impact of Evasion Attacks

This section presents the investigated benchmark electricity theft detectors along with their network parameters. Then, it discusses the influence of evasion attacks on the performance of the benchmark detectors in all attack settings.

A. Benchmark Detectors

To ensure that the conducted comparative analysis is comprehensive, we adopt six different machine learning-based detectors with a wide variety of properties, including shallow/deep, static/dynamic, or supervised/unsupervised. The benchmark detectors are based on ARIMA, 1-class SVM, 2-class SVM, feed forward, LSTM, and AAE.

1) Shallow Detectors: Shallow detectors utilize shallow machine learning-based methods. Thus, they do not fully capture the patterns within the electricity readings. The 1-class SVM classifier is static, whereas ARIMA is dynamic; both are trained on benign data only since they are anomaly detectors. The 2-class SVM is static and supervised since it uses benign and malicious data for training and testing.

2) Deep Detectors: Since deep detectors utilize deep learning-based methods, they can capture the complex patterns in the data. The feed forward-based classifier presents a deep structure where information flows in a forward direction, unlike the LSTM model where information cycles through loops. With the use of hidden layers, it learns informative features from the data and captures the complex patterns. However, it is still static and does not fully capture the temporal correlations in the electricity time series consumption data. The LSTM is an RNN-based classifier that captures the sequential patterns and temporal correlations within the data using hidden layers with LSTM cells accompanied with control gates. The AAE-based anomaly detector is trained only on benign data patterns to detect deviations in the test samples with the help of an encoder and detector along with an attentive layer.

3) Detection Decision: The detection decision of the anomaly detectors is made based on a threshold value ψ that separates benign and malicious samples. The shallow anomaly detectors use a minimum mean squared error (MSE) cost function to predict the future reading. Whenever the MSE exceeds ψ, the detector detects a malicious reading in the testing stage. The AAE compares the reconstruction error to ψ, which is determined by the median of the interquartile range (IQR) of the receiver operating characteristic (ROC) curve. If the score is above ψ, the sample is malicious with $y =$ '1'; else, the sample is benign with $y =$ '0'. For the supervised detectors trained on both classes, the output layer has two neurons, each defining a class where $y =$ '1' and $y =$ '0' labels denote malicious and benign classes, respectively.

B. Network Parameters

To get the best out of each detector, we adopt a grid-search hyperparameter optimization algorithm, where hyperparameters are tuned sequentially [48]. For the shallow detectors, we select the possible moving average and differencing degree values from {0, 1, 2, 3} for ARIMA. For the SVM models, kernel and gamma values are selected from {Linear, Sigmoid, rbf} and {scale, auto}, respectively. For the 2-class SVM, the regularization values are selected from {1, 10, 100}. For the deep detectors, the hyperparameters and values are: number of layers (in AAE L_{A}, LSTM L_{M}, and feed forward L_{F}): $\mathcal{L}_{(.)}$ = {2, 4, 6, 8}, number of neurons (in AAE N_{A}, LSTM N_{M}, and feed forward N_{F}): $\mathcal{N}_{(.)} = \{100, 200, 300, 500\}$, optimizer O: $\mathcal{O}$ = {Adam, Adamax, SGD}, dropout rate B: $\mathcal{B} = \{0, 0.2, 0.4, 0.5\}$, weight constraints G: $\mathcal{G} = \{0, 1, 3, 5\}$, hidden activation function A_H: $\mathcal{A}_{\text{H}}$ = {ReLU, Sigmoid, Linear}, and output activation functions A_O: $\mathcal{A}_{\text{O}}$ = {Softmax, Sigmoid}.

C. Experimental Results

1) Evasion Parameters: As discussed in Section III-D, it turns out that the ideal ε values range from 0.7 to 0.9 and from 0.4 to 0.6 when the operator utilizes shallow and deep detectors, respectively. k values range from 8 to 12 and from 4 to 8 when the operator utilizes shallow and deep detectors, respectively. Evasion parameters' values with deep detectors are lower than with shallow detectors since deep detectors are capable of extracting temporal correlations and complex patterns between the readings. Thus, deep detectors are more capable of detecting adversarial samples generated even with lower ε values, whereas shallow detectors notice adversarial samples only when the ε values are relatively high.

2) Attack Magnitude: When attackers perform traditional cyberattacks, the average daily decrease in electricity reports is around 5 kWh. Specifically, on average, applying $f_1(\cdot)$, $f_2(\cdot)$, $f_3(\cdot)$, $f_4(\cdot)$, $f_5(\cdot)$, and $f_6(\cdot)$ decrease the reported daily consumption by 6.7 kWh, 8.4 kWh, 4.8 kWh, 0.4 kWh, 9.8 kWh, and 0 kWh, respectively. However, applying the evasion attacks reduce the reported electricity consumption by 6.7 kWh, on average. Specifically, applying FGSM, BIM, AA, C&W, NNP, and NND lead to an average daily decrease in the reported electricity consumption of 4.8 kWh, 5.3 kWh, 5.9 kWh, 6.2 kWh, 8.5 kWh, and 9.7 kWh, respectively. Next, we study the impact of both attack types on the detection performance.

3) Network Parameters: We utilize Keras sequential API for training, with 100 epochs and a batch size of 100. The initial parameter values are kept as default. After running the optimization algorithm, the optimal moving average and differencing degree are 0 and 1, respectively, for ARIMA. For the SVM models, the kernel is sigmoid and gamma is scale, whereas the regularization parameter is 1.0 for the 2-class SVM model. The optimal feed forward network parameters turn out to be 8 layers with 300 neurons, Adamax optimizer, 0.2 dropout rate, weight constraint of 3, and Sigmoid hidden and output activation function. The optimal LSTM network parameters are 6 layers with 500 cells, Adam optimizer, no dropout rate, weight constraint of 5, ReLU and Softmax hidden and output activation functions, respectively. The AAE network parameters are $(500, 300, 200)$ LSTM cells in the 3 encoding layers and $(200, 300, 500)$ cells in the 3 decoding layers, SGD optimizer, 0.2 dropout rate, weight constraint of 1, and Sigmoid for the hidden and output activation function.

4) Threshold Values: As discussed in Section IV-B, the optimal threshold values ψ turn out to be 0.56, 0.45, and 0.51 for the ARIMA, 1-class SVM, and AAE, respectively.

5) Performance Metrics: To analyze the performance of the detectors, we produce a confusion matrix where we denote correctly identified malicious and benign readings as true positive (TP) and true negative (TN), respectively. Incorrectly identified benign and malicious readings are denoted as false positive (FP) and false negative (FN). Hence, we determine the detection and false alarm rates as $\text{DR} = \text{TP}/(\text{TP} + \text{FN})$ and $\text{FA} = \text{FP}/(\text{FP} + \text{TN})$, respectively.

6) Detection Performance: Table II reports the performance of the benchmark detectors when they are subject to traditional and evasion attacks using the white, gray, and black-box settings. The performance is reported according to the attack injection levels discussed in Section II-C. Specifically, the impact of evasion attacks on benchmark detectors is as follows.

- In the white-box setting, the average performance deterioration is $6.3 - 6.7\%$, $13.9 - 14.9\%$, $23 - 24.6\%$, and $33.4 - 35.8\%$ with 25%, 50%, 75%, and 100% of the proposed evasion attacks, respectively. With the benchmark evasion attacks, lower decrease rates of $4.5-5.3\%$, $10-12\%$, $16.9-20\%$, and $24.7-29.2\%$ are observed with 25%, 50%, 75%, and 100% of injection, respectively.
- In the gray-box setting, the average performance deterioration is $3.9 - 4.4\%$, $9.3 - 10.4\%$, $16 - 17.9\%$, and $24 - 26.9\%$ with 25%, 50%, 75%, and 100% of the proposed evasion attacks, respectively. Using the benchmark evasion attacks, we observe lower deterioration rates of $2.3-3.1\%$, $5.5-7.4\%$, $9.8-13\%$, and $15.1-19.8\%$ with 25%, 50%, 75%, and 100% of injection, respectively.
- In the perfect-match black-box setting, the attacker and operator adopt the same detector type where the attackers model's parameters are obtained using the substitute dataset. The average performance deterioration is $3 - 3.4\%$, $7.2 - 8.3\%$, $12.6 - 14.5\%$, and $19.2 - 22.2\%$ with 25%, 50%, 75%, and 100% of the proposed evasion attacks, respectively. Using the benchmark evasion attacks, we observe lower average deterioration rates of $1.2-2.1\%$, $3.4-5.2\%$, $6.4-9.7\%$, and $10.4-15.1\%$ with 25%, 50%, 75%, and 100% of injection, respectively.

Overall, attacks in white-box settings are the strongest since attackers have full access to the detector and dataset. Using partial access in gray-box settings, the average impact of evasion attacks is reduced by 2.2%, 4.5%, 6.7%, and 9% with 25%, 50%, 75%, and 100% of evasion injection, respectively, compared to white-box settings. Since attackers in black-box settings do not have access to the model's parameters nor the dataset, the attack impact is minimized by 3.2%, 6.6%, 10.1%, and 13.7% with 25%, 50%, 75%, and 100% of evasion injection, respectively, compared to white-box settings. Compared to benchmark evasion attacks, the proposed evasion attacks are stronger by up to $6.5-8.7\%$, $7.1-8.8\%$, and $7-8.7\%$ in the white, gray, and black-box settings, respectively, as they create adversarial samples via small dynamic unbounded ε based on the target and surrounding readings. Deep detectors are 4% more robust than shallow ones against evasion attacks.

V. Robust Electricity Theft Detection

Since existing state-of-the-art electricity theft detectors significantly suffer from evasion attacks, we propose a robust detector that maintains its detection performance even in the presence of high levels of evasion attacks in the toughest setting. The proposed detector combines deep neural networks in a sequential ensemble learning manner, which is an approach that extracts distinctive features by handling blocks in series where the output of each block is carried out to the next block, which boosts the detection performance [27].

A. Proposed Ensemble Learning Structure

This section presents the building blocks of the proposed sequential ensemble learning-based robust detector. As shown in Figure 3 and Algorithm 1, the detector places the input layer, followed by four blocks, namely, an RNN-based AAE, convolutional-recurrent part with one-dimensional convolutional, max-pooling, and additional LSTM layers, followed by fully connected layers and an output layer. The rationale behind this sequence is to differentiate between benign and malicious samples by capturing the complex patterns, temporal correlations, distinctive features, and top relevant features among them. This is achieved via the following sequence.

1) Attentive Autoencoder Block: The AAE block contains an encoder and decoder. Both have L_{A} hidden LSTM layers with N_{A} LSTM cells in each layer separated by an attentive layer. In the encoder's side, the LSTM's input is the reported reading $\boldsymbol{x}$ and the time series vector is encoded into a hidden state. The attentive layer receives the output from the encoder's side and the hidden state from the decoder so that distinct scores and weights are allocated to each time step where higher importance is assigned to the more contributing

TABLE II
IMPACT OF EVASION ATTACKS ON BENCHMARK DETECTORS (%)

			White-box Evasion Percentage					Gray-box Evasion Percentage					Black-box Evasion Percentage				
Attack	**Model**	**Metric**	**0**	**25**	**50**	**75**	**100**	**0**	**25**	**50**	**75**	**100**	**0**	**25**	**50**	**75**	**100**
FGSM	**ARIMA**	**DR**	85.3	80.4	74.3	67.1	58.7	85.3	82.5	78.7	73.8	67.9	85.3	83.6	80.9	77.2	72.6
		FA	14.5	19.3	25.2	32.3	40.5	14.5	17.2	21.0	25.8	31.7	14.5	16.2	18.8	22.4	27.0
	1-class	**DR**	87.6	82.9	77.1	70.1	61.9	87.6	85.1	81.5	76.8	71.1	87.6	86.1	83.6	80.2	75.9
	SVM	**FA**	12.7	17.3	23.0	29.9	37.9	12.7	15.2	18.7	23.2	28.8	12.7	14.2	16.6	19.9	24.1
	2-class	**DR**	89.2	84.6	78.8	71.9	63.9	89.2	86.8	83.4	78.9	73.4	89.2	87.8	85.5	82.2	78.0
	SVM	**FA**	10.2	14.7	20.3	27.1	35.0	10.2	12.5	15.8	20.2	25.6	10.2	11.5	13.7	16.8	20.9
	Feed	**DR**	90.8	86.4	80.9	74.3	66.5	90.8	88.6	85.4	81.1	75.8	90.8	89.7	87.6	84.6	80.7
	forward	**FA**	9.3	13.7	19.2	25.9	33.7	9.3	11.5	14.7	18.9	24.2	9.3	10.4	12.4	15.4	19.3
	LSTM	**DR**	91.5	87.2	81.7	75.1	67.4	91.5	89.5	86.4	82.3	77.2	91.5	90.6	88.8	86.1	82.5
		FA	7.0	11.2	16.5	23.0	30.6	7.0	8.9	11.8	15.7	20.6	7.0	7.8	9.5	12.2	15.8
	AAE	**DR**	94.1	90.0	84.8	78.5	71.1	94.1	92.4	89.7	86.0	81.3	94.1	93.4	91.7	89.1	85.6
		FA	5.2	9.4	14.7	21.1	28.6	5.2	7.0	9.8	13.6	18.4	5.2	6.0	7.7	10.3	13.8
BIM	**ARIMA**	**DR**	85.3	79.8	73.0	65.0	55.7	85.3	81.9	77.4	71.8	65.0	85.3	82.9	79.5	75.1	69.7
		FA	14.5	19.8	26.3	34.1	43.1	14.5	17.7	22.0	27.5	34.1	14.5	16.6	19.7	23.9	29.1
	1-class	**DR**	87.6	82.4	76.0	68.4	59.5	87.6	84.6	80.4	75.1	68.7	87.6	85.6	82.6	78.5	73.4
	SVM	**FA**	12.7	17.8	24.1	31.7	40.5	12.7	15.7	19.8	25.1	31.6	12.7	14.7	17.8	21.9	27.0
	2-class	**DR**	89.2	84.2	78.0	70.5	61.8	89.2	86.4	82.5	77.5	71.3	89.2	87.4	84.6	80.7	75.8
	SVM	**FA**	10.2	15.1	21.2	28.6	37.2	10.2	12.9	16.7	21.7	27.8	10.2	11.9	14.6	18.3	23.1
	Feed	**DR**	90.8	86.0	79.9	72.6	64.1	90.8	88.2	84.4	79.5	73.5	90.8	89.3	86.8	83.3	78.8
	forward	**FA**	9.3	14.0	19.9	27.0	35.3	9.3	11.8	15.4	20.1	25.9	9.3	10.7	13.2	16.7	21.2
	LSTM	**DR**	91.5	86.9	81.1	74.1	65.9	91.5	89.2	85.8	81.3	75.7	91.5	90.3	88.1	84.9	80.7
		FA	7.0	11.6	17.4	24.5	32.8	7.0	9.3	12.7	17.3	23.0	7.0	8.2	10.4	13.7	18.0
	AAE	**DR**	94.1	89.8	84.3	77.6	69.7	94.1	92.2	89.2	85.1	79.9	94.1	93.2	91.3	88.4	84.4
		FA	5.2	9.6	15.2	22.0	30.0	5.2	7.2	10.3	14.5	19.8	5.2	6.2	8.2	11.2	15.2
AA	**ARIMA**	**DR**	85.3	79.5	72.5	64.2	54.4	85.3	81.8	77.1	71.1	64.1	85.3	82.8	79.1	74.4	68.6
		FA	14.5	20.0	26.7	34.8	44.1	14.5	17.9	22.5	28.3	35.3	14.5	16.9	20.3	24.8	30.5
	1-class	**DR**	87.6	82.2	75.6	67.7	58.4	87.6	84.4	80.0	74.45	67.7	87.6	85.4	82.2	77.8	72.3
	SVM	**FA**	12.7	18.0	24.4	32.3	41.4	12.7	15.8	20.1	25.5	32.2	12.7	14.8	18.1	22.4	27.7
	2-class	**DR**	89.2	84.0	77.6	69.9	60.9	89.2	86.2	82.1	76.7	70.2	89.2	87.3	84.2	80.1	74.9
	SVM	**FA**	10.2	15.3	21.7	29.4	38.4	10.2	13.2	17.3	22.6	29.1	10.2	12.1	15.1	19.3	24.5
	Feed	**DR**	90.8	85.7	79.4	71.8	62.9	90.8	87.9	83.8	78.6	72.2	90.8	98.1	86.2	82.3	77.4
	forward	**FA**	9.3	14.3	20.5	28.0	36.8	9.3	12.1	16.1	21.3	27.6	9.3	11.0	13.8	17.7	22.7
	LSTM	**DR**	91.5	86.6	80.5	73.2	64.6	91.5	88.9	85.3	80.4	74.4	91.5	90.1	87.5	83.9	79.3
		FA	7.0	11.8	18.0	25.4	34.1	7.0	9.6	13.3	18.2	24.3	7.0	8.5	11.1	14.7	19.4
	AAE	**DR**	94.1	89.4	83.5	76.4	68	94.1	91.9	88.5	83.9	78.2	94.1	92.9	90.5	87.1	82.6
		FA	5.2	9.8	15.7	22.9	31.3	5.2	7.5	10.9	15.4	21.1	5.2	6.5	8.8	12.1	16.5
C&W	**ARIMA**	**DR**	85.3	79.4	72.1	63.4	53.4	85.3	81.6	76.7	70.5	63.1	85.3	82.6	78.7	73.6	67.4
		FA	14.5	20.2	27.2	35.5	45.2	14.5	18.1	23.0	29.2	36.6	14.5	17.1	20.8	25.7	31.8
	1-class	**DR**	87.6	82.1	75.3	67.1	57.5	87.6	84.2	79.6	73.8	66.7	87.6	85.2	81.7	77.0	71.2
	SVM	**FA**	12.7	18.1	24.8	32.9	42.3	12.7	15.9	20.3	25.9	32.8	12.7	14.9	18.3	22.8	28.4
	2-class	**DR**	89.2	83.9	77.3	69.3	60.0	89.2	86.0	81.6	75.9	69.0	89.2	87.1	83.9	79.5	74.0
	SVM	**FA**	10.2	15.6	22.3	30.3	39.6	10.2	13.4	17.8	23.5	30.4	10.2	12.4	15.7	20.2	25.9
	Feed	**DR**	90.8	85.5	78.9	71.0	61.8	90.8	87.7	83.3	77.7	70.9	90.8	88.8	85.6	81.3	75.9
	forward	**FA**	9.3	14.6	21.2	29.1	38.4	9.3	12.4	16.8	22.4	29.3	9.3	11.3	14.4	18.7	24.1
	LSTM	**DR**	91.5	86.4	80.5	72.3	63.3	91.5	88.7	84.7	79.5	73.1	91.5	89.8	86.9	82.9	77.8
		FA	7.0	12.1	18.6	26.4	35.5	7.0	9.8	13.8	19.1	25.6	7.0	8.8	11.7	15.7	20.8
	AAE	**DR**	94.1	89.1	82.8	75.2	66.3	94.1	91.5	87.7	82.7	76.5	94.1	92.5	89.7	85.8	80.8
		FA	5.2	10.1	16.3	23.8	32.6	5.2	7.7	11.4	16.3	22.4	5.2	6.7	9.3	13.0	17.8
NNP	**ARIMA**	**DR**	85.3	78.4	70.0	60.2	48.9	85.3	80.5	74.3	66.7	57.8	85.3	81.6	76.6	70.3	62.7
		FA	14.5	21.2	29.4	39.0	50.0	14.5	19.0	24.8	31.9	40.4	14.5	18.1	22.9	29.0	36.4
	1-class	**DR**	87.6	81.2	73.3	64.0	53.3	87.6	83.4	77.9	71.1	62.9	87.6	84.3	79.8	74.1	67.1
	SVM	**FA**	12.7	19.0	26.8	36.0	46.6	12.7	16.9	22.4	29.3	37.5	12.7	15.8	20.1	25.7	32.5
	2-class	**DR**	89.2	83.0	75.4	66.4	56.0	89.2	85.1	79.7	72.9	64.8	89.2	86.2	82.0	76.5	69.8
	SVM	**FA**	10.2	16.5	24.2	33.4	44.0	10.2	14.4	19.9	26.8	35.0	10.2	13.4	17.8	23.5	30.4
	Feed	**DR**	90.8	84.7	77.2	68.3	58.0	90.8	86.9	81.6	75.0	67.1	90.8	88.0	83.9	78.6	72.1
	forward	**FA**	9.3	15.5	23.1	32.2	42.7	9.3	13.3	18.6	25.2	33.1	9.3	12.3	16.6	22.1	28.8
	LSTM	**DR**	91.5	85.5	78.1	69.3	59.1	91.5	87.8	82.8	76.5	68.8	91.5	88.9	85.1	80.1	73.9
		FA	7.0	12.9	20.3	29.1	39.3	7.0	10.6	15.6	21.9	29.5	7.0	9.5	13.2	18.1	24.2
	AAE	**DR**	94.1	88.4	81.3	72.8	62.9	94.1	90.8	86.2	80.3	73.1	94.1	91.8	88.3	83.6	77.7
		FA	5.2	11.0	18.2	26.8	36.8	5.2	8.6	13.3	19.3	26.6	5.2	7.6	11.2	16.0	22.0
NND	**ARIMA**	**DR**	85.3	77.9	68.9	58.4	46.4	85.3	80.0	73.2	64.8	54.9	85.3	81.1	75.4	68.2	59.6
		FA	14.5	21.7	30.4	40.7	52.5	14.5	19.5	26.1	34.3	44.0	14.5	18.6	24.1	31.0	39.3
	1-class	**DR**	87.6	80.7	72.3	62.4	50.9	87.6	82.8	76.5	68.7	59.3	87.6	83.9	78.7	72.1	64.1
	SVM	**FA**	12.7	19.5	27.9	37.9	49.4	12.7	17.3	23.5	31.2	40.4	12.7	16.4	21.5	28.0	35.9
	2-class	**DR**	89.2	82.5	74.2	64.4	53.1	89.2	84.6	78.5	70.8	61.6	89.2	85.6	80.6	74.1	66.2
	SVM	**FA**	10.2	16.9	25.1	34.8	46.0	10.2	14.7	20.7	28.3	37.4	10.2	13.7	18.7	25.1	32.9
	Feed	**DR**	90.8	84.3	76.3	66.8	55.7	90.8	86.5	80.6	73.2	64.3	90.8	87.6	82.9	76.8	69.3
	forward	**FA**	9.3	15.9	24.1	33.8	45.0	9.3	13.8	19.8	27.3	36.4	9.3	12.6	17.3	23.5	31.1
	LSTM	**DR**	91.5	85.2	77.4	68.1	57.3	91.5	87.5	82.0	75.0	66.5	91.5	88.6	84.3	78.5	71.3
		FA	7.0	13.4	21.4	30.9	41.9	7.0	11.1	16.7	23.8	32.4	7.0	10.1	14.6	20.5	27.9
	AAE	**DR**	94.1	87.9	80.2	71.0	60.3	94.1	90.3	85.0	78.2	69.9	94.1	91.3	87.1	81.4	74.3
		FA	5.2	11.3	18.9	28.0	38.6	5.2	8.9	14.1	20.8	29.0	5.2	8.0	12.2	17.8	24.8

time steps towards the desirable output [26]. The input to the decoder is the reconstructed output and the attentive layer's output.

At time t, a state c_t is presented by an LSTM cell and outputs a hidden state h_t. In the encoder's side, there are three gates controlling the access to an LSTM cell, namely, input

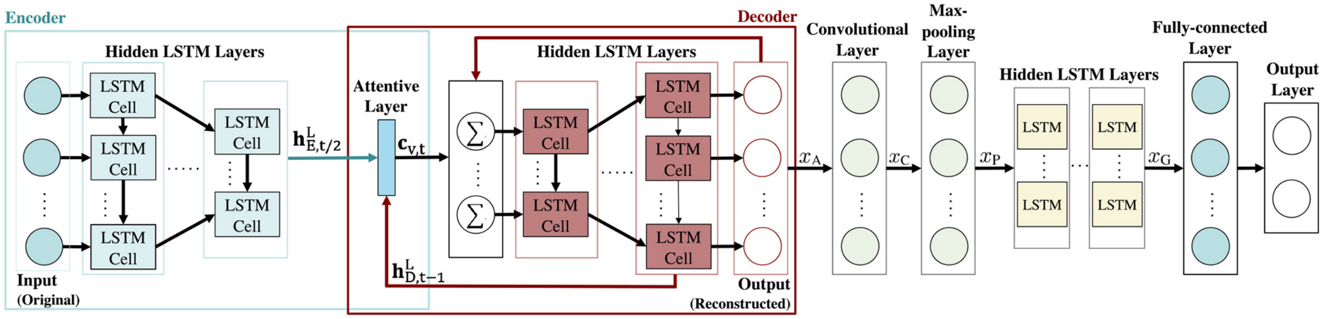


Fig. 3. Illustration of the proposed robust detector.

$i_{E,t}$, output $o_{E,t}$, and forget $f_{E,t}$ gates. In the decoder's side, the control gates are input $i_{D,t}$, output $o_{D,t}$, and forget $f_{D,t}$. The reading $\boldsymbol{x}_t$ is fed into an LSTM cell along with the hidden state of the previous cell of the same layer ($h_{E,t-1}$ and $h_{D,t-1}$ for the encoder and decoder's sides, respectively) and the state of the previous cell ($c_{E,t-1}$ and $c_{D,t-1}$ in the encoder and decoder's sides, respectively). In particular,

- $\boldsymbol{i}^l_{E/D,t} = \varphi(\boldsymbol{W}^l_i\boldsymbol{x}^l_t + \boldsymbol{U}^l_i\boldsymbol{h}^l_{E/D,t-1} + \boldsymbol{V}^l_i\boldsymbol{c}^l_{E/D,t-1} + \boldsymbol{b}^l_i)$
- $\boldsymbol{f}^l_{E/D,t} = \varphi(\boldsymbol{W}^l_f\boldsymbol{x}^l_t + \boldsymbol{U}^l_f\boldsymbol{h}^l_{E/D,t-1} + \boldsymbol{V}^l_f\boldsymbol{c}^l_{E/D,t-1} + \boldsymbol{b}^l_f)$
- $\boldsymbol{c}^l_{E/D,t} = f^l_{E/D,t}c^l_{E/D,t-1} + i^l_{E/D,t}\tanh(\boldsymbol{W}^l_c\boldsymbol{x}^l_t + \boldsymbol{U}^l_c\boldsymbol{h}^l_{E/D,t-1} + \boldsymbol{b}^l_c)$
- $\boldsymbol{o}^l_{E/D,t} = \varphi(\boldsymbol{W}^l_o\boldsymbol{x}^l_t + \boldsymbol{U}^l_o\boldsymbol{h}^l_{E/D,t-1} + \boldsymbol{V}^l_o\boldsymbol{c}^l_{E/D,t} + \boldsymbol{b}^l_o)$
- $\boldsymbol{h}^l_{E/D,t} = \boldsymbol{o}^l_{E/D,t}\tanh(\boldsymbol{c}^l_{E/D,t})$.

$\boldsymbol{h}^{L_A}_{E,t}$ and $\boldsymbol{h}^{L_A}_{D,t-1}$ denote the hidden states that the attentive layer receives. The context vector $\boldsymbol{c}_{v,t}$, that the attentive layer outputs, is achieved by an alignment scoring function $\boldsymbol{m}$, softmax function $\boldsymbol{s}$, and multiplication defined in lines 14 - 16 of Algorithm 1. In the decoder's side, the hidden layer acquires the concatenation $\sum(\boldsymbol{c}_{v,t}, x_A)$, where x_A denotes the reconstructed output. Hence, in this first block, as the AAE operates on the original data, it learns benign customers' behavioral patterns during the reconstruction process. The LSTM layers capture the temporal correlations within the original readings and handle the vanishing/exploding gradient problem while learning temporal correlations within the long intervals. The attentive layer assigns higher importance to time steps that contribute more towards the desired outcome.

2) Convolutional-Recurrent Block: In this second block, the AAE's output x_A is fed into a convolutional layer with learnable filters. These filters have a small size and are utilized to perform further feature extraction from x_A with strong dependency patterns. Convolution is applied on x_A such that the output of the convolutional layer is denoted by $x_C = x_A F_c + b_c$, where F_c depicts the learnable square filter matrix and b_c is the bias. Then, the max-pooling layer compresses the convolutional layer's output to capture the top relevant features. Hence, max-pooling is applied to x_C such that $x_P = \text{maxpool}(x_C)$. This way, the convolutional part serves as a trainable feature extractor. Then, the supplementary LSTM layers are added to apprehend more hidden features and temporal correlations from the top relevant extracted features.

3) Fully Connected and Output Blocks: In this third block, the output of the convolutional-recurrent part x_G is reshaped by the fully connected layer for making a final decision at the output layer. The fourth block represents the output layer that has two neurons defining two classes with $y =$ '1' and $y =$ '0' labels for the malicious and benign classes, respectively.

B. Robust Detector Training

In Algorithm 1, we use an iterative gradient descent optimization algorithm for training where X_{TR} is split into equal-sized M mini-batches. For I iterations, feed forward and back propagation are executed. In feed forward, the output vector is computed by passing training samples in M mini-batches through the layers. In back propagation, the gradient of the cost function given the weights of the network are calculated [48]. The gradients are used to update the iterations' weights and biases. The detection decision is made based on the MSE as discussed in Section IV-A3.

C. Experimental Results

The experimental settings of the proposed and benchmark detectors are the same in terms of the datasets, network and evasion parameters selection approach, and threshold value ψ.

1) Evasion Parameters: As discussed in Section III-D, it turns out that the ideal ε and k values vary from 0.2 to 0.4 and from 2 to 4, respectively, when the operator utilizes the proposed detector. This means that the proposed detector is capable of detecting adversarial samples even with smaller ε values, compared to the benchmark detectors.

2) Network Parameters: The AAE part and the supplementary LSTM layers have the same parameters discussed in Section IV-C3 when it comes to the number of layers and neurons. On the convolutional side, the used optimizer is SGD and the hidden activation is ReLU. The fully connected layer has 500 neurons. The rest of the optimized parameters are: Adam optimizer, 0.2 dropout rate, weight constraint of 1, ReLU for hidden and output activation functions.

3) Detection Performance: Table III presents the impact of evasion attacks on the proposed detector using the cases discussed in Section II-C. The impact is summarised as follows.

- In the white-box setting, the average performance deterioration is stable with only $0.3-0.4\%$, $0.8-1\%$, $1.5-1.9\%$, and $2.5 - 3\%$ at 25%, 50%, 75%, and 100% of the proposed strong evasion attacks, respectively. Using the benchmark evasion attacks, the performance decrease is $< 0.1 - 0.2\%$, $0.3 - 0.6\%$, $0.5 - 1.0\%$, and $0.8 - 1.6\%$ at 25%, 50%, 75%, and 100% injection, respectively.

Algorithm 1: Training of the Robust Detector

```
Input Data: $X_{\mathrm{TR}}$
Initialization: Weights $\boldsymbol{U}^{l}_{(\cdot)}$, $\boldsymbol{W}^{l}_{(\cdot)}$, $\boldsymbol{V}^{l}_{(\cdot)}$, and bias $\boldsymbol{b}^{l}_{(\cdot)}$ $\forall l$, $\boldsymbol{h}^{L}_{\mathrm{D},t-1}$ and $x_{\mathrm{A}}$
while not converged do
    for each training sample $\boldsymbol{x}$ do
        Feed forward
        Encoder:
        for each hidden layer $l = 1, \ldots, L/2$ do
            for each time step $t$ do
                The values of
                $\boldsymbol{i}^{l}_{\mathrm{E},t}$, $\boldsymbol{f}^{l}_{\mathrm{E},t}$, $\boldsymbol{c}^{l}_{\mathrm{E},t}$, $\boldsymbol{o}^{l}_{\mathrm{E},t}$, and $\boldsymbol{h}^{l}_{\mathrm{E},t}$
                are presented in Section V.A.1
                Attentive Layer:
                if $l = L/2$ then
                    $\boldsymbol{m} = \Gamma(\boldsymbol{h}^{L/2}_{\mathrm{E},t}, \boldsymbol{h}^{L}_{\mathrm{D},t-1})$ with feed forward model $\Gamma$
                    $\boldsymbol{s} = \exp(\boldsymbol{m}) / \sum_{|\boldsymbol{m}|} \exp(\boldsymbol{m})$
                    $\boldsymbol{c}_{\mathrm{v},t} = \sum_{T} \boldsymbol{s} \times \boldsymbol{h}^{L/2}_{\mathrm{E},t}$.
                end
            end
            $\boldsymbol{h}'^{l} = \boldsymbol{h}^{l}_{\mathrm{E},t}$, $\boldsymbol{c}'^{l} = \boldsymbol{c}^{l}_{\mathrm{E},t}$.
        end
        $\check{\boldsymbol{x}} = \sum(\boldsymbol{c}_{\mathrm{v},t}, x_{\mathrm{A}})$
        Decoder:
        The decoder hidden and cell states at initial time step are equal to $\boldsymbol{h}'^{l}$ and $\boldsymbol{c}'^{l}$
        for each hidden layer $l = L/2 + 1, \ldots, L$ do
            for each time step $t$ do
                The values of
                $\boldsymbol{i}^{l}_{\mathrm{D},t}$, $\boldsymbol{f}^{l}_{\mathrm{D},t}$, $\boldsymbol{c}^{l}_{\mathrm{D},t}$, $\boldsymbol{o}^{l}_{\mathrm{D},t}$, and $\boldsymbol{h}^{l}_{\mathrm{D},t}$
                are presented in Section V.A.1
            end
        end
        AAE output: $x_{\mathrm{A}}$
        Convolutional-recurrent:
        $x_{\mathrm{C}} = x_{\mathrm{A}} F_c + b_c$
        $x_{\mathrm{P}} = \mathrm{maxpool}(x_{\mathrm{C}})$
        $o_t^{l_{\mathrm{G}}-1} = x_{\mathrm{P}}$ for $l_{\mathrm{G}} = 1$
        for each recurrent layer $l_G$ do
            for each time step $t$ do
                The values of
                $\boldsymbol{i}_t^{l_{\mathrm{G}}}$, $\boldsymbol{f}_t^{l_{\mathrm{G}}}$, $\boldsymbol{c}_t^{l_{\mathrm{G}}}$, $\boldsymbol{o}_t^{l_{\mathrm{G}}}$, and $\boldsymbol{h}_t^{l_{\mathrm{G}}}$
                are presented in Section V.A.1
            end
        end
        Convolutional-recurrent output: $x_{\mathrm{G}} = o_t^{L_{\mathrm{G}}}$
        Fully connected Layer:
        Compute: $z^{l}(x_{\mathrm{G}}) = W^{l} \sigma(x_{\mathrm{G}}) + b^{l}$
        Back propagation: Compute:
        $\nabla_{\boldsymbol{U}^{l(.)}_{(.)}} C(x)$, $\nabla_{\boldsymbol{V}^{l(.)}_{(.)}} C(x)$, $\nabla_{\boldsymbol{W}^{l(.)}_{(.)}} C(x)$, and $\nabla_{\boldsymbol{b}^{l(.)}_{(.)}} C(x)$
    end
    Weight and bias update:
    $\boldsymbol{U}^{l(.)}_{(.)} = \boldsymbol{U}^{l(.)}_{(.)} - \frac{\eta}{K} \sum_x \nabla_{\boldsymbol{U}^{l(.)}_{(.)}} C(x)$
    $\boldsymbol{V}^{l(.)}_{(.)} = \boldsymbol{V}^{l(.)}_{(.)} - \frac{\eta}{K} \sum_x \nabla_{\boldsymbol{V}^{l(.)}_{(.)}} C(x)$
    $\boldsymbol{W}^{l(.)}_{(.)} = \boldsymbol{W}^{l(.)}_{(.)} - \frac{\eta}{K} \sum_x \nabla_{\boldsymbol{W}^{l(.)}_{(.)}} C(x)$
    $\boldsymbol{b}^{l(.)}_{(.)} = \boldsymbol{b}^{l(.)}_{(.)} - \frac{\eta}{K} \sum_x \nabla_{\boldsymbol{b}^{l(.)}_{(.)}} C(x)$
end
Output: Optimal $\boldsymbol{U}^{l(.)}_{(.)}$, $\boldsymbol{W}^{l(.)}_{(.)}$, $\boldsymbol{V}^{l(.)}_{(.)}$, and $\boldsymbol{b}^{l(.)}_{(.)}$ $\forall l$.
```

- In the gray-box setting, the average performance deterioration is stable with only $0.2 - 0.3\%$, $0.6 - 0.7\%$, $1.1 - 1.3\%$, and $1.8 - 2.1\%$ at 25%, 50%, 75%, and 100% of the strong evasion injection, respectively. The average performance deterioration is $< 0.1\%$, $< 0.1 - 0.4\%$, $0.3 - 0.8\%$, and $0.5 - 1.4\%$ at 25%, 50%, 75%, and 100% of the benchmark evasion attacks, respectively.
- In the black-box perfect-match setting, despite the presence of the strong proposed evasion attacks, minor deterioration rates of $0.2 - 0.3\%$, $0.5 - 0.7\%$, $0.8 - 1.1\%$, and $1.3 - 1.7\%$ are observed at 25%, 50%, 75%, and 100% evasion injection, respectively. Similarly, the average performance deterioration is $< 0.1\%$, $0.1 - 0.3\%$, $0.2 - 0.5\%$, and $0.4 - 0.9\%$ at 25%, 50%, 75%, and 100%, of the benchmark evasion attacks, respectively.

Despite the presence of traditional and strong evasion cyberattacks, the proposed detector presents an average improvement of $24 - 32.8\%$, $14.2 - 24.8\%$, $10 - 20.4\%$ in white, gray, and black-box settings, respectively, compared to the benchmark detectors with 100% of evasion percentage. Furthermore, with the strongest setting (white-box), attack type (NND), and evasion level (100%), the proposed detector maintains a stable performance that deteriorates only by 3%. The proposed model is robust as its detection is not exclusive to a limited set of attacks as it presents an unsupervised anomaly detector that is only trained on comprehensive benign datasets. During testing, it detects cyberattacks since they present deviation from the learned benign patterns. Hence, it is robust against other zero-day (unseen) attacks. Additionally, the proposed detector offers a stable performance even in white-box settings where attackers have full knowledge about the defense mechanism. This means that it is robust against other attacks including adaptive attacks that are created after a defense has been specified, where the attacker takes advantage of the knowledge about the defense [49].

4) Detection of Combined Evasion Attacks: In Tables II and III, we study the impact of each evasion attack type on the detectors separately. However, it is possible that attackers might launch more than one type of evasion attack at once. Hence, Table IV shows the detectors' performance when the adversarial samples are generated using a combination of the five types of evasion attacks during the same test. For example, the 25% evasion level contains 5% of each of the five evasion attacks. With all evasion levels, the performance of the shallow and deep detectors deteriorates further by $4.2 - 4.7\%$ and $3.1 - 3.6\%$, respectively, compared to the average deterioration when the attacks are launched separately. However, the proposed detector is able to maintain a similar performance that decreases by only $0.3 - 0.8\%$, which means that it is robust even when the test set contains a combination of different traditional and evasion attack functions.

5) Mismatch Black-Box Case: In the previous sections, we studied the impact of evasion attacks in a black-box setting where the attacker and utility company employ the same detection scheme (perfect-match) with different parameters. However, this may not always be the case. Hence, in Table V, we report the DRs in cases where the attacker uses a different detection scheme than the employed one (mismatch) using multiple possibilities. We conclude that evasion attacks still lead to higher degradation in detection than traditional attacks. Also, the proposed evasion attacks do not require the detector type perfect-match case to achieve high performance

TABLE III
IMPACT OF EVASION ATTACKS ON THE PROPOSED DETECTOR (%)

		White-box Evasion Percentage					Gray-box Evasion Percentage					Black-box Evasion Percentage				
Attack	**Metric**	**0**	**25**	**50**	**75**	**100**	**0**	**25**	**50**	**75**	**100**	**0**	**25**	**50**	**75**	**100**
FGSM	**DR**	96.2	96.2	96.0	95.8	95.5	96.2	96.2	96.1	95.9	95.7	96.2	96.2	96.1	96.0	95.8
	FA	2.3	2.3	2.6	2.8	3.1	2.3	2.3	2.5	2.6	2.9	2.3	2.3	2.4	2.5	2.7
BIM	**DR**	96.2	96.1	95.8	95.5	95.0	96.2	96.1	96.0	95.7	95.3	96.2	96.2	96.1	95.8	95.5
	FA	2.3	2.4	2.7	3.1	3.5	2.3	2.3	2.5	2.9	3.3	2.3	2.3	2.4	2.7	3.0
AA	**DR**	96.2	96.1	95.7	95.3	94.8	96.2	96.1	95.9	95.6	95.1	96.2	96.2	96.0	95.7	95.4
	FA	2.3	2.4	2.8	3.2	3.7	2.3	2.4	2.6	3.0	3.6	2.3	2.3	2.5	2.8	3.1
C&W	**DR**	96.2	96.0	95.6	95.2	94.6	96.2	96.1	95.8	95.4	94.9	96.2	96.1	95.9	95.7	95.3
	FA	2.3	2.5	2.9	3.4	3.9	2.3	2.4	2.6	3.2	3.7	2.3	2.4	2.6	2.9	3.2
NNP	**DR**	96.2	95.9	95.5	94.9	94.1	96.2	96.0	95.6	95.1	94.4	96.2	96.0	95.7	95.3	94.9
	FA	2.3	2.7	3.3	4.1	5.1	2.3	2.5	2.9	3.4	4.1	2.3	2.5	2.8	3.2	3.6
NND	**DR**	96.2	95.8	95.2	94.3	93.1	96.2	95.9	95.5	94.9	94.1	96.2	95.9	95.5	95.0	94.5
	FA	2.3	2.7	3.3	4.1	5.2	2.3	2.6	3.0	3.6	4.3	2.3	2.6	3.0	3.5	4.0

TABLE IV
IMPACT OF COMBINED EVASION ATTACKS ON THE DETECTORS (%)

		White-box Evasion Percentage					Gray-box Evasion Percentage					Black-box Evasion Percentage				
Attack	**Metric**	**0**	**25**	**50**	**75**	**100**	**0**	**25**	**50**	**75**	**100**	**0**	**25**	**50**	**75**	**100**
ARIMA	**DR**	85.3	74.8	67.3	58.6	48.2	85.3	76.9	71.7	65.2	57.4	85.3	78.0	74.0	68.7	62.4
	FA	14.5	24.8	31.9	40.7	50.6	14.5	22.6	27.7	34.0	41.6	14.5	21.7	25.6	30.6	37.0
1-class SVM	**DR**	87.6	77.6	70.6	62.2	52.3	87.6	79.8	74.9	68.8	61.6	87.6	80.8	77.0	72.0	66.2
	FA	12.7	22.6	29.6	37.9	47.5	12.7	20.4	25.2	31.3	38.5	12.7	19.4	23.1	28.0	33.9
2-class SVM	**DR**	89.2	79.5	72.6	64.2	54.8	89.2	81.7	77.1	71.1	63.9	89.2	82.7	79.2	74.5	68.6
	FA	10.2	19.9	26.8	35.0	44.5	10.2	17.8	22.4	28.3	35.4	10.2	16.8	20.2	24.9	30.8
Feed forward	**DR**	90.8	82.0	75.5	67.3	57.9	90.8	84.3	79.9	74.0	67.1	90.8	87.0	82.1	77.8	72.2
	FA	9.3	18.0	24.7	32.9	42.3	9.3	15.9	20.3	26.0	33.0	9.3	14.8	18.0	22.4	28.0
LSTM	**DR**	91.5	83.1	76.4	68.5	59.4	91.5	85.4	81.2	75.8	69.1	91.5	86.5	83.5	79.3	74.1
	FA	7.0	15.4	22.0	30.0	39.2	7.0	13.1	17.4	22.6	29.4	7.0	12.0	15.0	19.0	24.3
AEA	**DR**	94.1	85.9	79.7	72.0	63.0	94.1	88.3	84.5	79.4	73.1	94.1	89.4	86.7	82.6	77.5
	FA	5.2	13.3	19.7	27.4	36.4	5.2	10.9	14.8	20.0	26.3	5.2	9.9	12.8	16.7	21.8
Proposed	**DR**	96.2	95.4	94.9	94.5	93.7	96.2	95.6	95.3	94.9	94.3	96.2	95.6	95.5	95.2	94.7
	FA	2.3	3.2	3.6	4.2	4.8	2.3	2.8	3.2	3.5	4.2	2.3	2.7	3.0	3.3	3.8

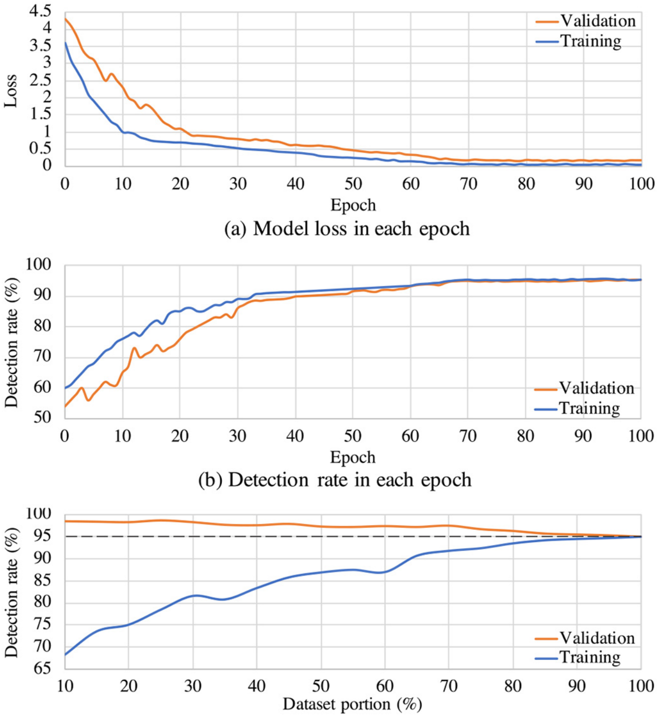
TABLE V
UTILITY/ATTACKER MISMATCH DR PERFORMANCE (%)

Attack model	**2-class SVM**	**1-class SVM**	**Feed forward**	**ARIMA**	**Pro-posed**	**AAE**
Utility model	**1-class SVM**	**2-class SVM**	**ARIMA**	**Feed forward**	**AAE**	**Pro-posed**
FGSM	82.7	83.8	78.2	85.9	90.1	96.1
BIM	80.1	81.5	75.1	83.8	88.7	95.8
AA	78.9	80.6	73.9	82.4	86.8	95.7
C&W	77.7	79.7	72.6	80.8	84.9	95.6
NNP	71.0	73.1	65.6	74.6	80.5	95.1
NND	67.8	69.4	62.4	71.8	76.9	94.7

degradation (results are very close to the ones reported in Table I for the NNP and NND attacks). Finally, the proposed detector achieves a stable detection performance in both the mismatch cases and the perfect-match cases shown in Table II.

6) Other Defense Mechanisms: Besides the benchmark detectors, other adversarial defense mechanisms may be used against evasion attacks. For example, adversarial training [31] familiarizes the model with adversarial samples by augmenting the training set with such samples in each training loop. Certified defenses [32] output a certificate of robustness for two-layer networks. MagNet [33] trains a reformer network to move adversarial samples closer to the manifold of benign samples in white-box attack settings. In GAN-based frameworks [34], besides the detector, two additional models are trained in an adversarial setting, namely, a generative model to mimic the data distribution, and a discriminative model to differentiate between benign and malicious samples. However, such defense mechanisms require an additional step as well as computational resources besides the implemented detection scheme. Additionally, they require a specific assumption either about the network or attack settings.

Table VI compares the performance of the proposed detectors with relevant state-of-the-art adversarial defense mechanisms accompanied with the same architectures proposed by the authors. The reports are with 50% evasion injection level using a combination of the five types of evasion attacks. The addition of the adversarial defense mechanisms to the networks improves the detection performance compared to benchmark detectors by $1.1-2.3\%$, $1.6-2.7\%$, and $2.2-3\%$ in the white, gray, and black-box settings, respectively. Despite such slight improvements, our proposed detector still outperforms the state-of-the-art adversarial defense mechanisms by $12.3-17.8\%$, $7.5-13.2\%$, and $5.1-11.1\%$ in the white, gray, and black-box settings, respectively, reflecting its superiority to state-of-the-art adversarial defense mechanisms.

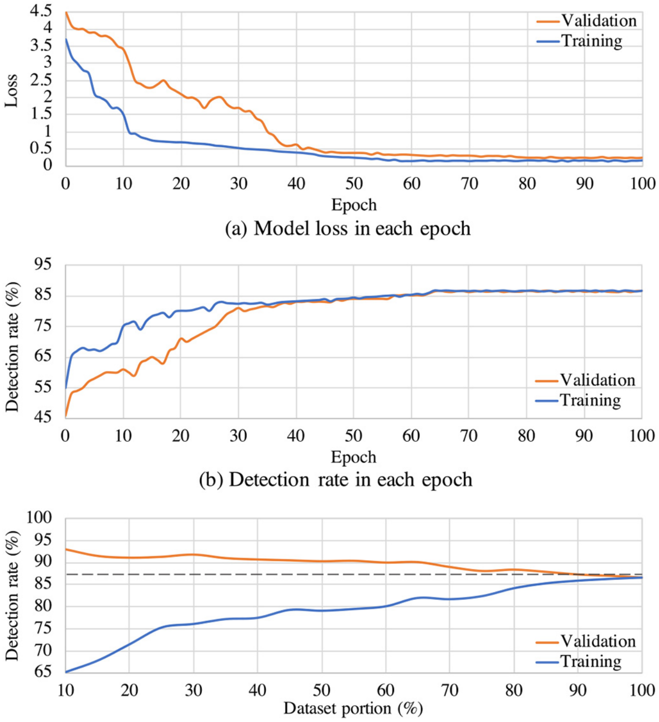


(a) Model loss in each epoch

(b) Detection rate in each epoch

(c) Detection rate with different dataset portions

Fig. 4. Learning curves of the proposed model.



(a) Model loss in each epoch

(b) Detection rate in each epoch

(c) Detection rate with different dataset portions

Fig. 5. Learning curves of a benchmark model (AAE).

TABLE VI
IMPACT OF EVASION ATTACKS ON ADVERSARIAL DEFENSE MECHANISMS (%)

		Attack Setting		
Defense	**Metric**	**White-box**	**Gray-box**	**Black-box**
Adversarial Training	**DR**	76.6	81.5	84.3
	FA	23.7	18.8	15.9
Certified Defense	**DR**	77.0	82.1	84.4
	FA	23.3	18.2	15.4
MagNet	**DR**	81.6	86.9	89.6
	FA	17.7	12.3	10.0
GAN	**DR**	82.0	87.2	89.7
	FA	17.4	12.3	9.8
Proposed	**DR**	94.9	95.3	95.5
	FA	3.6	3.2	3.0

TABLE VII
DR OF THE PROPOSED DETECTOR USING DIFFERENT TRAINING SETTINGS (%)

		Attack Setting		
Parameters	**Dataset Size**	**White-box**	**Gray-box**	**Black-box**
Default Parameters	$\mathbf{0.5}\lvert\boldsymbol{X}_{\mathrm{TR}}\rvert$	82.4	85.2	88.2
	$\mathbf{0.75}\lvert\boldsymbol{X}_{\mathrm{TR}}\rvert$	87.9	89.5	90.7
	$\lvert\boldsymbol{X}_{\mathrm{TR}}\rvert$	91.0	91.8	92.4
Tuned Parameters	$\mathbf{0.5}\lvert\boldsymbol{X}_{\mathrm{TR}}\rvert$	86.8	89.2	91.7
	$\mathbf{0.75}\lvert\boldsymbol{X}_{\mathrm{TR}}\rvert$	92.3	93.5	94.1
	$\lvert\boldsymbol{X}_{\mathrm{TR}}\rvert$	94.9	95.3	95.5

D. Model Analysis

This subsection analyzes how well the proposed model is trained. The analysis is based on the learning curves, percentage of utilized dataset portion, impact of hyperparameter tuning, and computational complexity.

1) Learning Curves: Figure 4 plots the training and validation sets learning curves of the proposed model in a white-box setting. Figure 4a illustrates how the model loss is decreasing as the number of epochs increases. Figure 4b illustrates how the DR is improving as the number of epochs increases. DR is reported herein to determine how well the model detects traditional and evasion attacks. The learning curves converge at epoch 70, offering stable DR of around 95% up until the last epoch, reflecting the maximum detection performance of the model with constant convergence rate. Figure 4c demonstrates the DR improvement level as the portion of the utilized dataset increases. The maximum DR is offered when the entire dataset is utilized. For comparison and to avoid repetition, in Figure 5, we plot the learning curves of the best-performing benchmark detector (AAE). The AAE model along with the rest of the benchmark models behave similar to the learning curves in Figure 4 with similar convergence rates, reflecting minimum model loss and maximum detection performance throughout the epochs.

2) Dataset Portion: Table VII reports the DR of the proposed detector when trained on different number of samples where $|\boldsymbol{X}_{\mathrm{TR}}|$ denotes the entire dataset size and $|\cdot|$ denotes the cardinality. When trained on 75% of the dataset, the proposed detector still offers a stable performance that is $1.3 - 2.6\%$ lower than training on the entire dataset.

3) Impact of Hyperparameter Tuning: Table VII also compares the DR of the proposed detector when trained on the default and tuned parameters. The default parameters are: SGD optimizer, no dropout rate and weight constraint, and Relu hidden and output activation function. The tuned parameters are discussed in Sections IV-C3 and V-C2. Tuning the parameters improves the DR by $3.1 - 4.4\%$.

4) Computational Complexity: The models are trained offline using the NVIDIA GeForce RTX 2070 hardware accelerator. The benchmark detectors take up to 3 hours to be trained. Adding defense mechanisms increases the training time by around 2 hours. Since our proposed detector places multiple blocks sequentially, the training time increases to 4.5 hours. Testing the detectors is done online and takes 2 seconds to make the decisions on individual readings. Utility companies may train the model on available datasets offline (not in real time) and run the detection scheme online during the billing period to apply penalties accordingly. Hence, training time is not a pressing issue herein.

VI. Conclusion

This work studied the impact of various traditional and complex evasion cyberattack types and levels on multiple machine learning-based electricity theft detectors using different attack settings. We proposed strong evasion attacks (NNP and NND) that create adversarial samples via small dynamic unbounded perturbation values based on the target and surrounding readings to fool the detector. We compared their impact to benchmark evasion attacks (FGSM, BIM, AA, and C&W) that apply constant bounded perturbation values generated using one target reading. The proposed attacks are stronger than benchmark ones by 8.7% and deteriorate the detection performance of benchmark detectors by 35.8%, 26.9%, and 22.2% in white, gray, and black-box settings, respectively. For performance enhancement, we proposed a robust sequential ensemble learning-based detector combining AAE, convolutional-recurrent, and fully connected neural networks, whose performance deteriorates only by $0.7-3\%$, $0.9-2.1\%$, and $0.4-1.7\%$ in white, gray, and black-box settings, respectively, with highest attack injection levels. Despite our solid results, maintaining a stable detection performance is challenging in white-box settings due to the attacker's knowledge, which can be further studied in future works.

References

[1] A. Takiddin, M. Ismail, and E. Serpedin, “Robust detection of electricity theft against evasion attacks in smart grids,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2021, pp. 1–6.

[2] P. Jokar, N. Arianpoo, and V. C. Leung, “Electricity theft detection in AMI using customers' consumption patterns,” *IEEE Trans. Smart Grid*, vol. 7, no. 1, pp. 216–226, Jan. 2016.

[3] V. B. Krishna, C. A. Gunter, and W. H. Sanders, “Evaluating detectors on optimal attack vectors that enable electricity theft and DER fraud,” *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 4, pp. 790–805, Aug. 2018.

[4] Y. Peng *et al.*, “Electricity theft detection in AMI based on clustering and local outlier factor,” *IEEE Access*, vol. 9, pp. 250–259, 2021.

[5] S. K. Singh, R. Bose, and A. Joshi, “PCA based electricity theft detection in advanced metering infrastructure,” in *Proc. 7th Int. Conf. Power Syst. (ICPS)*, 2017, pp. 441–445.

[6] V. B. Krishna, K. Lee, G. A. Weaver, R. K. Iyer, and W. H. Sanders, “F-DETA: A framework for detecting electricity theft attacks in smart grids,” in *Proc. 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, 2016, pp. 407–418.

[7] M. Wen, D. Yao, B. Li, and R. Lu, “State estimation based energy theft detection scheme with privacy preservation in smart grid,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[8] T. Murthy, N. Gopalan, and V. Ramachandran, “A Naïve Bayes classifier for detecting unusual customer consumption profiles in power distribution systems—APSPDCL,” in *Proc. 3rd Int. Conf. Inventive Syst. Control (ICISC)*, Coimbatore, India, 2019, pp. 673–678.

[9] R. Wu, L. Wang, and T. Hu, “AdaBoost-SVM for electrical theft detection and GRNN for stealing time periods identification,” in *Proc. Conf. IEEE Ind. Electr. Soc.*, Washington, DC, USA, 2018, pp. 3073–3078.

[10] Z. Yan and H. Wen, “Electricity theft detection base on extreme gradient boosting in AMI,” *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–9, Jan. 2021, doi: 10.1109/TIM.2020.3048784.

[11] R. Punmiya and S. Choe, “Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing,” *IEEE Trans. Smart Grid*, vol. 10, no. 2, pp. 2326–2329, Mar. 2019.

[12] S. Li, Y. Han, X. Yao, S. Yingchen, J. Wang, and Q. Zhao, “Electricity theft detection in power grids with deep learning and random forests,” *J. Electr. Comput. Eng.*, vol. 2019, Oct. 2019, Art. no. 4136874, doi: 10.1155/2019/4136874.

[13] S. O. Tehrani, M. H. Y. Moghaddam, and M. Asadi, “Decision tree based electricity theft detection in smart grid,” in *Proc. 4th Int. Conf. Smart City Internet Things Appl. (SCIOT)*, Nov. 2020, pp. 46–51.

[14] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra, “Decision tree and SVM-based data analytics for theft detection in smart grid,” *IEEE Trans. Ind. Informat.*, vol. 12, no. 3, pp. 1005–1016, Jun. 2016.

[15] V. Krishna, R. K. Iyer, and W. H. Sanders, ‘ARIMA-based modeling and validation of consumption readings in power grids,” in *Critical Information Infrastructures Security*. Cham, Switzerland: Springer Int., May 2016, pp. 199–210.

[16] A. Ullah, N. Javaid, O. Samuel, M. Imran, and M. Shoaib, “CNN and GRU based deep neural network for electricity theft detection to secure smart grid,” in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, 2020, pp. 1598–1602.

[17] M. N. Hasan, R. N. Toma, A.-A. Nahid, M. M. M. Islam, and J.-M. Kim, “Electricity theft detection in smart grid systems: A CNN-LSTM based approach,” *Energies*, vol. 12, no. 17, p. 3310, Aug. 2019.

[18] M. Nabil, M. Ismail, M. Mahmoud, M. Shahin, K. Qaraqe, and E. Serpedin, “Deep learning-based detection of electricity theft cyber-attacks in smart grid AMI networks,” in *Deep Learning Applications for Cyber Security*. Cham, Switzerland: Springer, 2019, pp. 73–102.

[19] M. Nabil, M. Mahmoud, M. Ismail, and E. Serpedin, “Deep recurrent electricity theft detection in AMI networks with evolutionary hyper-parameter tuning,” in *Proc. Int. Conf. Inter. Things*, Atlanta, GA, USA, 2019, pp. 1002–1008.

[20] Y. He, G. J. Mendis, and J. Wei, “Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism,” *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2505–2516, Sep. 2017.

[21] A. Takiddin, M. Ismail, M. Nabil, M. M. E. A. Mahmoud, and E. Serpedin, “Detecting electricity theft cyber-attacks in AMI networks using deep vector embeddings,” *IEEE Syst. J.*, vol. 15, no. 3, pp. 4189–4198, Sep. 2020.

[22] F. Shehzad, N. Javaid, A. Almogren, A. Ahmed, S. M. Gulfam, and A. Radwan, “A robust hybrid deep learning model for detection of non-technical losses to secure smart grids,” *IEEE Access*, vol. 9, pp. 128663–128678, 2021.

[23] I. U. Khan, N. Javeid, C. J. Taylor, K. A. A. Gamage, and X. Ma, “A stacked machine and deep learning-based approach for analysing electricity theft in smart grids,” *IEEE Trans. Smart Grid*, vol. 13, no. 2, pp. 1633–1644, Mar. 2022.

[24] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, “Deep autoencoder-based detection of electricity stealth cyberattacks in AMI networks,” in *Proc. Int. Symp. Signals, Circuits Syst. (ISSCS)*, 2021, pp. 1–6.

[25] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, “Variational autoencoder-based detection of electricity stealth cyber-attacks in AMI networks,” in *Proc. Eur. Signal Process. Conf. (EUSIPCO)*, Amsterdam, The Netherlands, 2021, pp. 1590–1594.

[26] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, “Deep autoencoder-based anomaly detection of electricity theft cyberattacks in smart grids,” *IEEE Syst. J.*, early access, Jan. 7, 2022, doi: 10.1109/JSYST.2021.3136683.

[27] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, “Robust electricity theft detection against data poisoning attacks in smart grids,” *IEEE Trans. Smart Grid*, vol. 12, no. 3, pp. 2675–2684, May 2021.

[28] H. Rao, Y. Bai, C. Zhang, D. He, and Y. Chen, “Adversarial example attack on electric power network security situation awareness,” in *Proc. Inf. Technol. Netw. Electron. Autom. Control Conf. (ITNEC)*, 2021, pp. 1394–1398.

[29] J. Tian, B. Wang, Z. Wang, K. Cao, J. Li, and M. Ozay, “Joint adversarial example and false data injection attacks for state estimation in power systems,” *IEEE Trans. Cybern.*, early access, Nov. 19, 2021, doi: 10.1109/TCYB.2021.3125345.

[30] G. R. Mode and K. A. Hoque, "Adversarial examples in deep learning for multivariate time series regression," in *Proc. IEEE Appl. Imagery Pattern Recognit. Workshop (AIPR)*, 2020, pp. 1–10.

[31] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," Mar. 2015, *arXiv:1412.6572v3*.

[32] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," Oct. 2020, *arXiv:1801.09344v2*.

[33] D. Meng and H. Chen, "MagNet: A two-pronged defense against adversarial examples," in *Proc. ACM Conf. Comput. Commun. Security (CCS)*, Nov. 2017, pp. 135–147.

[34] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-Gan: Protecting classifiers against adversarial attacks using generative models," May 2018, *arXiv:1805.06605v2*.

[35] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," Feb. 2017, *arXiv:1607.02533v4*.

[36] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 2206–2216.

[37] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 39–57.

[38] "Irish Social Science Data Archive." [Online]. Available: https://tinyurl.com/us86zuaj (Accessed: Mar. 2022).

[39] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2008, pp. 1322–1328.

[40] "Smart-Grid Smart-City Customer Trial Data." [Online]. Available: https://tinyurl.com/9wftuaf2 (Accessed: Mar. 2022).

[41] K. Ren, T. Zheng, Z. Qin, and X. Liu, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346–360, Mar. 2020.

[42] A. Nazemi and P. Fieguth, "Potential adversarial samples for white-box attacks," Dec. 2019, *arXiv:1912.06409*.

[43] M. Juuti, B. G. Atli, and N. Asokan, "Making targeted black-box evasion attacks effective and efficient," in *Proc. ACM Workshop Artif. Intell. Security*, London, U.K., 2019, pp. 83–94.

[44] C. Guo, J. R. Gardner, Y. You, A. G. Wilson, and K. Q. Weinberger, "Simple black-box adversarial attacks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 2484–2493.

[45] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," Sep. 2019, *arXiv:1706.06083v4*.

[46] F. Croce and M. Hein, "Minimally distorted adversarial examples with a fast adaptive boundary attack," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 2196–2205.

[47] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: A query-efficient black-box adversarial attack via random search," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 484–501.

[48] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[49] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, Stockholm, Sweden, 2018, pp. 274–283.