On the Impact of Model Tolerance in Power Grid Anomaly Detection Systems

Srinidhi Madabhushi and Rinku Dewri

University of Denver, Denver CO 80208, USA
nidhi.madabhushi@du.edu, rdewri@cs.du.edu

Abstract. Rapid development in deep learning-based detection systems for numerous industrial applications has opened opportunities to apply them in power grids. A consumer's power consumption can be monitored to recognize any anomalous behavior in their household. When building such detection systems, evaluating their robustness to adversarial samples is critical. It has been shown that when we provide adversarial samples to deep learning models, they falsely classify instances, even when the perturbation or noise added to the original data is very small. On the other hand, these models should be able to detect attack instances correctly and raise few to no false alarms. While this expectation can be difficult to attain, we are allowed to choose a threshold that decides the extent to which the detection and false alarm rates are compromised. To this end, we explore the threshold selection problem for state-of-the-art deep learning-based detection models such that it can recognize attack instances. We show that selecting a threshold is challenging, and even if an appropriate threshold is chosen, the tolerance of a model to adversarial samples can still leave avenues for an attack to be successful.

Keywords: Anomaly Detection · Deep Learning · Model Sensitivity · Power Consumption · Threshold Selection

1 Introduction

Industrial control systems support critical national infrastructure that are essential for managing various industries like electricity generation and distribution, water treatment and supply, oil and gas production and many more. Disruption of such infrastructure at any time can lead to serious effects on the society and can impact the safety and economy of a nation. A large scale attack on a power grid allows adversaries to take control and operate other industrial control systems as well. Moreover, the attack surface for a power grid has increased over the past decade with the advent of IoT devices. As these devices are designed with security as an afterthought, they become easy targets for attackers. By controlling IoT devices of consumers, an attacker can regulate the power demand that can lead to grid failures. Therefore, using anomaly detection and monitoring systems to detect anomalous power consumption can help in identifying such cases in advance, facilitating grid operators to take necessary action. Machine

learning and deep learning methods are proposed for many anomaly detection applications, and neural networks are well known for their ability to learn patterns, making them applicable for time series data. However, it is possible for adversaries to modify the input data to deceive the model. Adversarial attacks on such models were first shown in image applications where perturbed images of animals and traffic signs were mis-classified, when small perturbations are introduced in the images [6,16]. Previous work also showed that by perturbing the time series data in small amounts, a deep neural network can still be deceived for time series classification applications [5,14].

In this paper, we study how susceptible deep learning-based anomaly detection models are to adversarial samples that manipulate the power demand through compromised consumer devices. First, we provide a simple yet easily generalizable method to create adversarial samples that is parameterized by the frequency and strength of the attack (Section 3). This allows us to compare any generic anomaly detection system against another in a black box setting. Second, we provide a comparative assessment of the detection efficiency of three state-of-the-art deep learning models for power consumption anomaly detection under different levels of attack aggressiveness. Choosing a right cut-off threshold is critical for such systems because they differentiate anomalous values from the normal ones. Hence, third, we demonstrate through a detailed exercise that it can be challenging to choose a threshold that can detect the various attack scenarios, and provide a low false alarm rate at the same time (Section 4). We show that a model may perform well for certain levels of attack aggressiveness, but is unable to provide coverage across all scenarios. Lastly, we delve deeper into the types of undetected attacks in each model, providing insights on the characteristics of the models and the trade-off between the model's noise tolerance levels and the attacker's wattage requirement (Section 5). Unfortunately, there always appears to be sufficient room for an attacker to bypass detection by adjusting their manipulation frequency and strength. We conclude the paper in Section 6 with references to future work.

2 Background and Related Work

In this section, we discuss demand manipulation attacks (MAD) and their effects on the power grid. We explain the role of anomaly detection algorithms for detecting MAD attacks. We further discuss existing literature for anomaly detection in the power consumption domain, and adversarial attacks demonstrated on deep learning models.

2.1 Demand Manipulation Attacks

Demand manipulation attacks, also known as MAD attacks, occur when an adversary manipulates the demand of the power grid from the utility side using consumer devices. These devices are assumed to be in a residential setting and are manipulated either directly by the attacker or by the consumer. Such attacks

can be performed by controlling a botnet of IoT devices that can manipulate the power demand much faster than the power plants can react [4]. When an attacker has access to various high wattage IoT devices, they can synchronously switch them on and off which leads to the disruption of the power grid [17]. The attacker can also influence the behavior of the consumers by sending false messages to fake maintenance shutdown alerts, suggesting the consumers to use appliances during peak consumption periods [15]. Such attacks have adverse effects on the power grid such as (i) frequency instability that can lead to a sudden generation tripping, or disrupting a grid re-start, (ii) line failures and cascading failures, and (iii) an increase in operating costs leading to the Independent System Operators (ISOs) having to purchase additional power in the form of reserve generators [17].

2.2 Anomaly Detection Mechanism

The power consumption data of a household that is collected from a smart meter provides an opportunity to detect sudden changes in the consumption as a result of a demand manipulation attack. Anomaly detection mechanisms are designed for detecting attacks and alerting the consumer and the power grid operator. This allows the power grid personnel to take necessary action to avoid potential damage to the power grid equipment. An anomaly detection mechanism consists of a prediction method followed by a scoring technique which provides a score specifying how anomalous the instance is. A thresholding mechanism is then used to provide a cut-off beyond which the instance will be flagged as an anomaly.

2.3 Related Work

A review for several anomaly detection systems proposed for power consumption data was conducted by Himeur et al. focusing on artificial intelligence-based (AI) models [8]. They categorized models based on different aspects like the type of algorithm and application. While this work covers different AI models, we focus on deep learning and neural network-based models. As power consumption data is a time series, long short term memory (LSTM) neural networks are predominantly used for time series prediction applications. Wang et al. proposed an LSTM-based detection model where the predictions are used to calculate anomaly scores for each time unit and two thresholds are used to mark anomalies in the data [19]. Clustering-based approaches like K-means can also be applied instead of scores to identify anomalous instances [2].

Neural networks are often combined with other techniques to capture events that can be detected by both models. While taking advantage of LSTM's ability to remember previous observations, it can be combined with traditional time series methods like ARIMA to model the non-linear components of the data [9]. Autoregression models (AR) can also be combined with feed forward neural networks (NN) to form a hybrid model called NNAR [3]. Kim et al. combined two neural networks, CNN and LSTM, for predicting power consumption data [10]. Other examples of two neural networks used in conjunction include combining

autoencoders with Online Sequential Extreme Learning Machine (OS-ELM) and LSTM [18,20].

Adversarial attacks on deep neural network models were demonstrated by Goodfellow et al. in a computer vision application [7]. Different adversarial attack generation methods on neural networks have been proposed including fast gradient sign method (FGSM) [7] and basic iterative method (BIM) [12] among many others. These methods are designed for a threat model where the adversary has full knowledge of their target model (white-box). However, due to the transferability of adversarial samples across models, these methods are effective for gray-box (knowledge limited to model structure) and black-box (can only query the model) threat models [16]. In the literature, different adversarial sample generation models have been compared by applying them to neural networks used for time series classification tasks, including power consumption applications [5]. This was also demonstrated for multivariate time series regression models specifically for CNNs, LSTMs and GRUs (Gated Recurrent Unit) [14].

In this work, we do not assume a target model and generate adversarial samples by incrementally increasing the power injection and proportion of perturbation. We then pass the generated adversarial samples to three detection models to assess their performance. While adversarial attacks have been demonstrated on detection systems used in power grids as well as other industrial control systems [13,21], we focus on exploring whether a detection system when tuned to detect such adversarial samples, succeeds or not.

3 Methodology

Anomaly detection systems that are semi-supervised are trained on normal data and when provided with new data, the model identifies anomalies based on how different the data is from what it learned. However, there may be cases where the data is anomalous, but the model is unable to detect it. The same applies to when the data is normal but the model identifies it as abnormal, because it is slightly different from what it has seen. In order to explore the research question of how well a model can avoid giving false classifications, we develop a methodology where different attack data are generated and tested on state-of-the-art models to evaluate their performance. The minute-level consumption values from the previous hour (60 minutes) represented by $d_{t-60}, d_{t-59}, ..., d_{t-2}, d_{t-1}$ are used to predict the power consumption at time t represented by d'_t. An anomaly score s_t is then calculated using the actual and predicted consumption values (d_t and d'_t). The threshold th is used to decide whether the consumption value at time t is an anomaly, by checking if the score s_t is greater than th.

3.1 Power Consumption Data

The data used in this paper is the individual household electric power consumption dataset obtained from the UCI Machine Learning Repository[1]. It consists

[1] https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption

of readings for nine attributes collected for every minute over a period of almost four years. As we focus only on power consumption, we use three attributes which consists of the date, time and global active power in kilowatt collected during the years 2007, 2008 and 2009. There are $25,979$ missing values in the data that are handled using linear interpolation which is particularly useful for time series. The training and validation data consists of power consumption values for the years 2007 and 2008, whereas the testing data consists of values for the year 2009. For predicting the power consumption at time t, the consumption values at times $t-1$, $t-2$, $t-3$,..., $t-60$ are passed as the input to the model. The training input for each neural network is a dataframe consisting of 60 columns that represent the previous 60 consumption values for each time unit and $920,100$ rows representing the inputs for timestamps for 21 months, starting from 2007-01-01 00:00:00 to 2008-09-30 23:59:00 with minute level sampling. The validation data consists of consumption values for three months starting from 2008-10-01 00:00:00 to 2008-12-31 23:59:00. All the consumption values for the year 2009 are used as test data.

3.2 Model Training

We select three prediction models from the literature, each representing a different type of neural network—multi layer perceptron (MLP) [3], long short term memory (LSTM) [2] and convolutional neural network LSTM (CNN-LSTM) [10]. We follow semi-supervised training in which the training data is considered to be normal and does not contain any anomalies. For each model, we use the same number of layers that the authors from the literature use. We tune the hyperparameters of the neural network by starting off with the values that the literature uses and going higher and lower than the suggested value in powers of two. We perform this search for the number of nodes, dropout, batch size and number of epochs. The search is controlled by the mean squared error and provides the model that has the least error on the validation data. The number of nodes per layer and other parameters for the final models are shown in Table 1. The layers are listed in the order of their placement in the architecture. There are two convolution layers for the CNN-LSTM model for which the parameters for each layer are given in the format *(number of filters, kernel size)*. The number of input features for each model is 60 and the output is a single value, the prediction. Mean squared error is used as the loss function, Adam [11] as the optimizer and TensorFlow [1] on the back-end.

3.3 Anomaly Score

An anomaly score gives the extent to which the data point should be considered as an anomaly. We use the score s_t adopted by Wang et al. [19], given as

$$s_t = \frac{|predicted_t - observed_t|}{avg_{i \in T}(|predicted_i - observed_i|)}, \tag{1}$$

Table 1. Model architecture for MLP, LSTM and CNN-LSTM

Parameter	MLP	LSTM	CNN-LSTM
Convolution layer	n/a	n/a	(64, 2), (64, 2)
LSTM layer	n/a	32, 32, 64	128
Dense layer	100	n/a	32, 64
Dropout	n/a	0.07, 0.03	n/a
Number of epochs	30	5	30
Batch size	2048	2048	4096

Table 2. Thresholds calculated for each model using the validation data with varying percentiles

Percentile	MLP	LSTM	CNN-LSTM
60.0	0.39	0.63	0.41
70.0	0.54	0.74	0.58
80.0	0.92	1.03	0.92
90.0	2.14	2.11	2.37
95.0	4.52	4.08	4.75
98.0	8.74	7.4	8.04
99.0	11.94	10.21	11.16
99.5	15.68	13.21	14.06
99.9	21.8	18.93	19.78
99.999	39.4	34.88	36.08

where $predicted_t$ is the prediction for consumption at time t, $observed_t$ is the observed consumption at time t and all previous times $\{1, 2, 3, ..., t-2, t-1\} \in T$.

3.4 Thresholding Mechanism

A threshold is a value that is applied to an anomaly score above which a point is flagged as an anomaly. We use a percentile-based approach to calculate ten different thresholds that vary by the percentile values. This lets us explore the threshold that fits best with the data. The percentile values used are 60, 70, 80, 90, 95, 98, 99, 99.5, 99.9 and 99.999 which starts from the strictest threshold and is loosened till the 99.999^{th} percentile. The threshold values calculated for each model are shown in Table 2. The percentiles are calculated using anomaly scores for the validation data.

3.5 Attack Profiles

An attack profile represents a dataset that consists of attack instances and is created by adding perturbations to the original dataset. We generate 2,000 attack profiles using a perturbation model that randomly chooses the time instances that are injected with extra wattage in the test data. The number of attack instances in the data depends on the proportion of perturbation ranging from 0.05 to 1.0 in steps of 0.05, giving a total of 20 proportions to choose from.

The perturbation that is added ranges from 1 kW to 100 kW in steps of 1 kW, giving 100 different perturbation values. We choose to limit at 100 kW as the injection is performed for a single household and it is unlikely to get a wattage as high as 100 kW. We extend the perturbations till 100 kW to study the model performance for increasing wattage. The following steps are used to generate the attack profiles.

1. For a proportion p, randomly choose the list of indices I (or timestamps) in the dataframe that will be treated as attack instances from the entire test dataset D.

$$I = random(n = length(D), r = length(D) \times p) \tag{2}$$

   where n is the total number of indices in the dataset D, r is the number of attack instances to select from D and $random$ chooses r values from 0 to n.
2. Add the perturbation ϵ to the chosen attack instances to create the attack profile D'.

$$D' = D[I] + \epsilon \tag{3}$$

3. Repeat steps 1 and 2 for each proportion $p \in 0.05, 0.1, 0.15,..., 0.95, 1.0$ and $\epsilon \in 1, 2, 3,..., 99, 100$.

4 Threshold Selection

Prediction-based anomaly detection systems require a threshold to be selected such that it is able to detect majority of the attack instances and give least number of false alarms at the same time. In this section, we explore different thresholds in order to choose a value that is able to perform well in both cases. The metrics for the threshold selection used are detection rate (DR) and false alarm rate (FAR). Detection rate measures the proportion of true anomalous instances that are correctly identified. False alarm rate measures the proportion of normal instances that are incorrectly classified as anomalies.

To make it easier for visual analysis, we categorize the combination of the amount of perturbation (injected power wattage) and the frequency of the attacks (percentage of attack instances) into nine attack configurations that represent the strength of the attack as shown in Figure 1. As both values get higher, it represents a more aggressive attack. We also overlay this grid on the plots to get a general overview of the performance of the metrics.

4.1 The Threshold Dilemma

The threshold dilemma represents the inability to choose a threshold based on the detection and false alarm rates in order to achieve the best performance for adversarial samples. It involves going back and forth with the threshold selection because there is a downside with one metric when a threshold is chosen using the

high perturbation less frequent	high perturbation moderately frequent	high perturbation more frequent
moderate perturbation less frequent	moderate perturbation moderately frequent	moderate perturbation more frequent
low perturbation less frequent	low perturbation moderately frequent	low perturbation more frequent

Fig. 1. Types of attacks varying by amount of perturbation and frequency of attacks

other metric. A chosen threshold must be able to detect most of the anomalies while keeping the false alarms low. This means it should have a high detection rate i.e. having a value closer to 1.0 and a low false alarm rate i.e. having a value closer to 0. The detection and false alarm rates are plotted for each threshold and each plot shows the performance across different attacks. Figure 2 shows the metrics when using four different thresholds that are calculated using the 60^{th}, 70^{th}, 80^{th} and 90^{th} percentile values (going from bottom to top). Though we calculate results for all the ten thresholds as described in Section 3.4, we choose to display only these four thresholds due to space constraints. The results for the other thresholds are predictable based on the patterns observed in the displayed plots. The x-axis represents the attack frequency that tells what proportion of attack instances are perturbed, whereas the y-axis represents the amount of perturbation or the power wattage that is injected to the chosen attack instances. As the models are trained using attack-free instances, it can be observed in the plots that the detection rates are invalid for all models where the percentage of attack instances or amount of perturbation is zero (i.e. no attack).

In general, the false alarm rates are the best for the highest threshold and the detection rates are the best for the least threshold. For all the models, we start off by choosing a threshold using a single metric and change the selection based on the performance of the other metric. We repeat this and go through all the thresholds which eventually causes a dilemma of what threshold is the best. We demonstrate this process in detail for the MLP model and generalize it for the LSTM and CNN-LSTM models.

We begin by choosing a threshold for the MLP model based on the performance results shown in Figure 2 . When selecting a threshold, it is preferred to have the least false alarm rate because it would be flustering to have false alarms raised too often. For this reason, we start by choosing the highest threshold of 2.14 calculated using the 90^{th} percentile. Note that the false alarm rate is the least for a 99.999^{th} percentile threshold, but we are only considering the displayed four thresholds for easy reference to the readers. When looking at the results for the 90^{th} percentile threshold, the model has low false alarm rates for majority of the attack configurations. But it is unable to maintain it as the attack frequency increases, thus failing in the right strip of the plot particularly when 80 to 100 percent of data is injected with any amount of wattage. If we are willing to accept the high false alarm rates for the higher proportions, the choice of threshold may seem fit. However, the detection rate for the same threshold

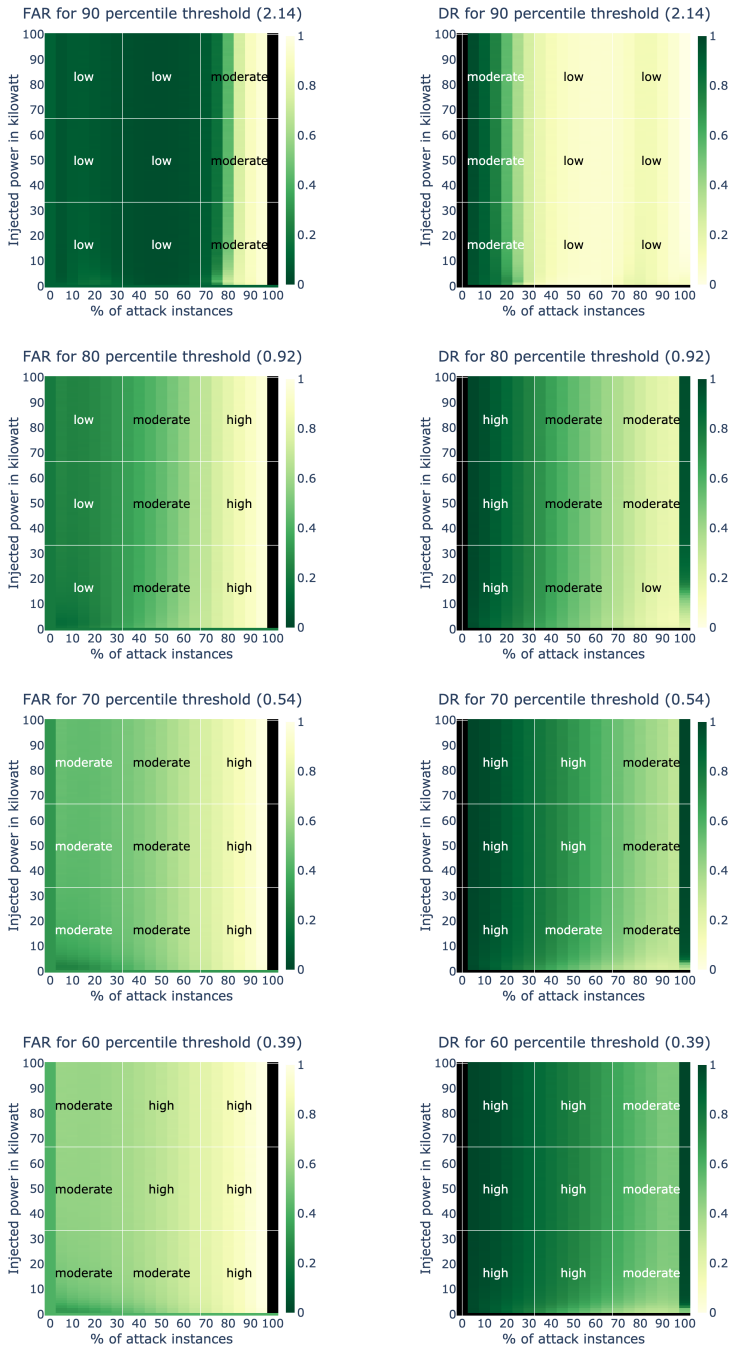

Fig. 2. False alarm rate (FAR) and detection rate (DR) for MLP with thresholds ranging from 60^{th} to 90^{th} percentile

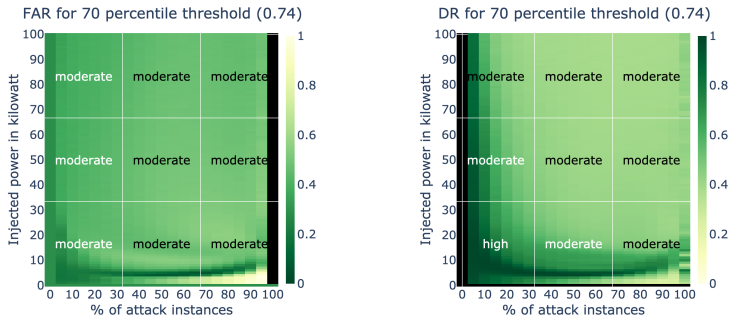

Fig. 3. False alarm rate and detection rate for LSTM for a selected threshold

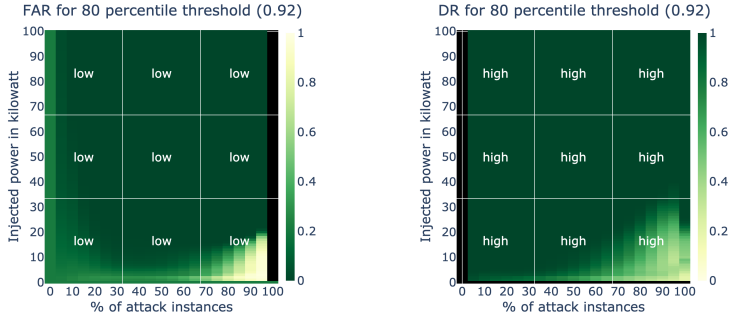

Fig. 4. False alarm rate and detection rate for CNN-LSTM for a selected threshold

of 90^{th} percentile (top-right of Figure 2) is poor even when 25% or more data is perturbed. Six out of nine attack configurations are almost never detected. In an attempt to improve the detection rate to the highest value possible, we will intuitively choose the least threshold of 60^{th} percentile (bottom-right of Figure 2), because it is able to detect majority of attacks across all attack zones compared to the others. However, when looking at the false alarm rates for the same threshold (bottom-left of Figure 2), the proportion of false alarms are between moderate and high for all attack configurations starting from the least to the highest possible perturbation and attack frequency. To improve the false alarm rates, we can increase the threshold to the highest unexplored threshold which is at 80^{th} percentile. There is an improvement in the false alarm rates throughout for this threshold compared to the 60^{th} percentile. While this seems to have given us a better threshold, the detection rates again are observed to be between moderate and low for attack frequencies greater than 50%. The only available threshold is at 70^{th} percentile, which provides an improvement for the detection rates. However, the false alarm rates have worsened in this case compared to the previous 80^{th} percentile threshold. This leads to a dilemma as to

which threshold must be chosen to balance both metrics. Even if a threshold is chosen, the compromise that occurs with either of the metrics leads to the conundrum again.

5 Model Tolerance and Impact

Similar to the MLP model, the behavior of both metrics to changing thresholds is the same for LSTM and CNN-LSTM models. As the threshold is increased from 60^{th} percentile to 90^{th} percentile, the false alarm rates improve and the detection rates deteriorate. Figures 3 and 4 shows both metrics for LSTM and CNN-LSTM models for a selected threshold. The decision to choose a threshold becomes challenging while trying to balance both the metrics for these models as well. Compared to all models, CNN-LSTM has high detection rates for most thresholds. However, the dilemma arises when deciding how much of the false alarm rates we are willing to compromise.

Through the above exercise, we observe that there exists a dilemma of what is to be chosen as the final threshold for the detection system irrespective of the neural network used. Pursuing the best performance for one metric often results in the other metric performing poorly. In this section, we discuss the characteristics and the prediction performance of each model to analyze their effects on the detection and false alarm rates. We also explore whether the inability of a threshold to detect certain attack configurations, can still lead to a successful and significant attack on the power grid.

Detection Model Characteristics A property that is common to all models is the inability to detect attacks in the bottom-right corner, which represents high frequency attacks with low perturbations. This could be because all the models start adapting to the small injections in the consumption that is performed to majority of the instances. The more the model starts seeing these small changes, the more it accepts it to be normal. However, the detection system is able to correctly identify sudden increase or decrease in the power consumption.

When we look at model-specific characteristics, the MLP model has linearly varying detection rates and false alarm rates based on the proportion of perturbation. As observed in Figure 2, if we choose 10% of instances to be perturbed, irrespective of the power injected, the MLP model has high detection rates throughout. On the other hand, the LSTM model follows an exponential shape for high detection rates as observed in Figure 3. It is able to detect all of small proportions and small power injections. For example, a perturbation value of 6 kW is consistently detected for any proportion. The CNN-LSTM model starts to fail detection in the bottom right corner as seen in Figure 4, and slowly propagates towards the left in an exponential shape. The detection rate at 100% perturbation proportion is different from the rest of the proportions for all thresholds in all models. This can be observed in Figures 2, 3 and 4 that there is a different behavior for only the 100% perturbation with progressively changing detection rates going from low to high for different perturbations.

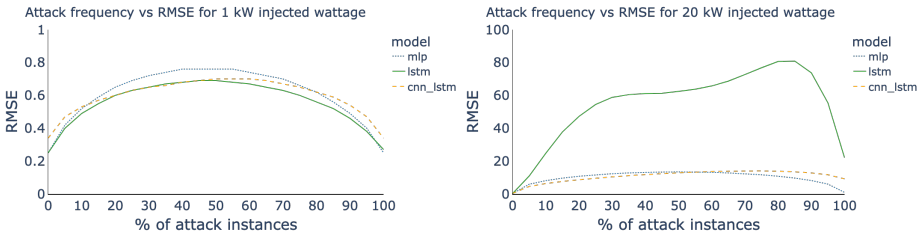

Fig. 5. Root mean squared error for different models when different proportions of attack instances are injected with 1 kW (left) and 20 kW (right)

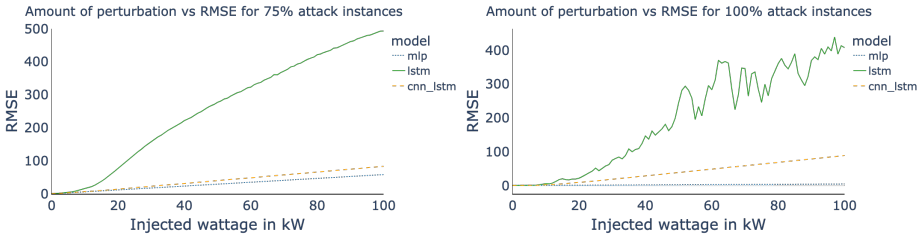

Fig. 6. Root mean squared error for different models when different values of power are injected for 75% (left) and 100% (right) of attack proportions

Model Prediction Performance The root mean squared error for the trained MLP, LSTM and CNN-LSTM models on validation data are 0.099, 0.114 and 0.145 respectively. In terms of predicting the power consumption values, MLP performs the best having the least error, followed by LSTM which has similar performance as MLP and lastly, CNN-LSTM. Figure 5 shows how the root mean squared error (RMSE) changes for each model as the percentage of attack instances increases. We can see that when the attack instances are perturbed with 1 kW increase in power, the RMSE values follow an inverted parabolic curve. This suggests that the prediction errors are lower when the proportion of attack instances are either very low or very high. This can be observed in the results from the previous section where the detection rates start becoming poor when there is a large proportion of attack instances. As the injected wattage is increased, MLP and CNN-LSTM models maintain the same shape, but the LSTM model starts to deviate from it. This can be observed in the plot for the 20 kW case, where the errors for LSTM model increase quickly, thus being sensitive to higher perturbation values. When we keep the percentage of attack instances constant and plot the RMSE values for increasing amount of injected wattage, the LSTM model again has a different behavior compared to the MLP and CNN-LSTM models. As seen in Figure 6, when the proportion of attack instances is 75%, the RMSE has increasing trend for all models. However, LSTM model increases very quickly for increasing wattage values. Whereas, for a 100%

proportion case, the RMSE values for LSTM increase and decrease for different values of power injections, while keeping the same increasing trend. This can be observed in Figure 3 for LSTM, where the detection rates fluctuate for a 100% attack proportion.

Successful Attacks on each Model Though the threshold selection might lead to an indecision, we select a value that balances both metrics for the purpose of discussing the impact of taking advantage of a model's sensitivity. If we are to choose the best threshold value for the MLP model, an 80^{th} percentile threshold with a value of 0.92 is able to give correct results, at least until 70% proportion of perturbed data. For the LSTM and CNN-LSTM models, the 70^{th} percentile and the 80^{th} percentile thresholds respectively, have a good balance between the detection rates and the false alarm rates. Now, we look into undetected attack configurations and decide whether the compromise made for selecting the threshold is negligible. If an adversary has to perform an attack on such a system that involves neural networks, an attack that is more frequent with low perturbations will be successfully obscured. The required wattage for carrying out a successful MadIoT attack to disrupt the power grid as proposed by Soltan et al. is 30 megawatt with 300 bots per megawatt [17]. This wattage can be injected by an adversary by controlling compromised IoT bots in the same geographical location. The number of bots required for the attack is even lower with high wattage devices. If an adversary has access to $3,000$ bots, the wattage per bot requirement would be 10kW for a successful attack. Similarly, with access to $6,000$ and $100,000$ bots, it would require 5 kW/bot and 0.3 kW/bot respectively.

When we look at the metrics for the chosen threshold of the MLP model which is 80^{th} percentile in Figure 2, the right vertical strip of the plot with proportions greater than 70% are not detected. Let us assume that the attacker has access to $3,000$ bots each contributing 10 kW of additional wattage. Given that for a single household, the detection is low for the MLP model, irrespective of the power injection for an 80% perturbation, the attack will be successful with access to 10 kW devices in $3,000$ households. With access to 20 kW or more per device and perturbing more than 30% of the consumption, the attacker can attain low detection rates with below 50% of the attack being detected in an LSTM model with the chosen threshold at 70^{th} percentile. As observed in the LSTM plot for 70^{th} percentile threshold in Figure 3, the detection rates are the least with lower injected wattage per device and more than 50% of the data perturbed. With this configuration, the attacker requires access to $6,000$ devices with each device contributing a lower wattage of 5 kW. If we go with the chosen 80^{th} percentile for the CNN-LSTM model, the attacker can again perform a successful attack by choosing devices under 30 kW and injecting it in more than 70% of the data as seen in Figure 4. The attack is possible with less than $3,000$ devices contributing more than 10 kW each. If we choose a higher threshold of 70^{th} percentile to combat this behavior, there is still a possibility for the adversary to use between 1 kW to 5 kW devices requiring $6,000$ to $9,000$ devices to carry out a successful attack.

In summary, the MLP model is easily vulnerable because of its exposure to a variety of small and large power injection attacks when 70% or more of instances are perturbed. LSTM model cannot detect higher power injection with moderate to high frequency attacks, and low power injection with high frequency attacks, with the latter being more feasible. Though an attacker can have a tough time staying undetected for majority of the attack configurations in a CNN-LSTM model, there is still a slight possibility for this to take place for lower power injections with high attack frequencies. Tightening or increasing the threshold for any of these models to improve the detection rates leads to the threshold dilemma as the false alarm rates will be significantly high. Therefore, going past this dilemma and choosing an appropriate threshold will still lead to successful attacks on the power grid as the attacker can work within the noise tolerance levels of a model to stay undetected.

6 Conclusion and Future Work

In this work, we train three state-of-the-art neural network models that are integrated into anomaly detection systems for power consumption data. We generate adversarial samples with increasing power wattage and attack frequency for the selected detection systems. We then demonstrate the threshold dilemma showing the inability to choose a single threshold that will be able to detect all power grid attacks. We discuss the model characteristics and the impact of the undetected attacks on the power grid. Using some of these insights, a possible future direction would be to investigate why neural network-based models fail when detecting attacks that have low power injection but high proportion of attack instances. As this is a common case observed in all models, it could be possible to offset that behavior by introducing a new parameter or model into the detection system to capture such instances. Though the metrics calculated for these cases have different behaviors visually, the errors for all models follow the same pattern except for LSTM. As LSTMs are commonly used for time series predictions, it would be interesting to look into why the model has increasing errors with high perturbation and proportion of attack instances.

References

1. Abadi, M., et al.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), `https://www.tensorflow.org/`, software available from tensorflow.org
2. Chahla, C., Snoussi, H., Merghem, L., Esseghir, M.: A deep learning approach for anomaly detection and prediction in power consumption data. Energy Efficiency **13**(8), 1633–1651 (2020)
3. Chou, J.S., Telaga, A.S.: Real-time detection of anomalous power consumption. Renewable and Sustainable Energy Reviews **33**, 400–411 (2014)
4. Dabrowski, A., Ullrich, J., Weippl, E.R.: Grid shock: Coordinated load-changing attacks on power grids: The non-smart power grid is vulnerable to cyber attacks

as well. In: 33rd Annual Computer Security Applications Conference. pp. 303–314 (2017)
5. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Adversarial attacks on deep neural networks for time series classification. In: 2019 International Joint Conference on Neural Networks. pp. 1–8. IEEE (2019)
6. Gnanasambandam, A., Sherman, A.M., Chan, S.H.: Optical adversarial attack. In: IEEE/CVF International Conference on Computer Vision. pp. 92–101 (2021)
7. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
8. Himeur, Y., Ghanem, K., Alsalemi, A., Bensaali, F., Amira, A.: Artificial intelligence based anomaly detection of energy consumption in buildings: A review, current trends and new perspectives. Applied Energy **287**, 116601 (2021)
9. Hollingsworth, K., Rouse, K., Cho, J., Harris, A., Sartipi, M., Sozer, S., Enevoldson, B.: Energy anomaly detection with forecasting and deep learning. In: 2018 IEEE International Conference on Big Data. pp. 4921–4925. IEEE (2018)
10. Kim, T.Y., Cho, S.B.: Predicting the household power consumption using CNN-LSTM hybrid networks. In: International Conference on Intelligent Data Engineering and Automated Learning. pp. 481–490. Springer (2018)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: Artificial Intelligence Safety and Security, pp. 99–112. Chapman and Hall/CRC (2018)
13. Li, J., Yang, Y., Sun, J.S.: Exploiting vulnerabilities of deep learning-based energy theft detection in ami through adversarial attacks. arXiv preprint arXiv:2010.09212 (2020)
14. Mode, G.R., Hoque, K.A.: Adversarial examples in deep learning for multivariate time series regression. In: 2020 IEEE Applied Imagery Pattern Recognition Workshop. pp. 1–10. IEEE (2020)
15. Raman, G., Peng, J.C.H., Rahwan, T.: Manipulating residents' behavior to attack the urban power distribution system. IEEE Transactions on Industrial Informatics **15**(10), 5575–5587 (2019)
16. Ren, K., Zheng, T., Qin, Z., Liu, X.: Adversarial attacks and defenses in deep learning. Engineering **6**(3), 346–360 (2020)
17. Soltan, S., Mittal, P., Poor, H.V.: BlackIoT: IoT botnet of high wattage devices can disrupt the power grid. In: 27th USENIX Security Symposium. pp. 15–32 (2018)
18. Tsukada, M., Kondo, M., Matsutani, H.: A neural network-based on-device learning anomaly detector for edge devices. IEEE Transactions on Computers **69**(7), 1027–1044 (2020)
19. Wang, X., Zhao, T., Liu, H., He, R.: Power consumption predicting and anomaly detection based on long short-term memory neural network. In: 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis. pp. 487–491. IEEE (2019)
20. Weng, Y., Zhang, N., Xia, C.: Multi-agent-based unsupervised detection of energy consumption anomalies on smart campus. IEEE Access **7**, 2169–2178 (2018)
21. Zizzo, G., Hankin, C., Maffeis, S., Jones, K.: Adversarial attacks on time-series intrusion detection for industrial control systems. In: 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications. pp. 899–910. IEEE (2020)