information

Article

Detecting and Mitigating Adversarial Examples in Regression Tasks: A Photovoltaic Power Generation Forecasting Case Study

Everton Jose Santana [1], Ricardo Petri Silva [2], Bruno Bogaz Zarpelão [3,*] and Sylvio Barbon Junior [3]

[1] Department of Software Engineering, Pontifical Catholic University of Paraná, Londrina 86067000, Brazil; everton.santana@pucpr.br
[2] Department of Electrical Engineering, State University of Londrina, Londrina 86057970, Brazil; petri@uel.br
[3] Department of Computer Science, State University of Londrina, Londrina 86057970, Brazil; barbon@uel.br
* Correspondence: brunozarpelao@uel.br

Abstract: With data collected by Internet of Things sensors, deep learning (DL) models can forecast the generation capacity of photovoltaic (PV) power plants. This functionality is especially relevant for PV power operators and users as PV plants exhibit irregular behavior related to environmental conditions. However, DL models are vulnerable to adversarial examples, which may lead to increased predictive error and wrong operational decisions. This work proposes a new scheme to detect adversarial examples and mitigate their impact on DL forecasting models. This approach is based on one-class classifiers and features extracted from the data inputted to the forecasting models. Tests were performed using data collected from a real-world PV power plant along with adversarial samples generated by the Fast Gradient Sign Method under multiple attack patterns and magnitudes. One-class Support Vector Machine and Local Outlier Factor were evaluated as detectors of attacks to Long-Short Term Memory and Temporal Convolutional Network forecasting models. According to the results, the proposed scheme showed a high capability of detecting adversarial samples with an average F1-score close to 90%. Moreover, the detection and mitigation approach strongly reduced the prediction error increase caused by adversarial samples.

Keywords: intelligent cyber-physical systems; adversarial machine learning; smart grid; security; photovoltaic generation forecast

check for updates

Citation: Santana, E.J.; Silva, R.P.; Zarpelão, B.B.; Barbon Junior, S. Detecting and Mitigating Adversarial Examples in Regression Tasks: A Photovoltaic Power Generation Forecasting Case Study. *Information* **2021**, *12*, 394. https://doi.org/10.3390/info12100394

Academic Editor: Willy Susilo

Received: 1 September 2021
Accepted: 18 September 2021
Published: 26 September 2021

1. Introduction

Wind and solar energy are the most acceptable and promising resources of renewable energy due to their potential and availability. In particular, photovoltaic (PV) facilities have experienced an enormous technological advance over the last few years, exploiting the advantages of using recent architectures such as Internet of Things (IoT) and Cloud Computing [1]. IoT sensors can collect variables such as weather conditions, system temperature, and generated power in PV power plants, which may indicate faults and contribute to the understanding of the plant's generation capacity. By accessing this information online, operators can be prepared for promptly handling unexpected events and variations [2].

Machine learning (ML) is an important building block for successfully integrating PV power plants into smart grids. ML algorithms can underpin solutions to analyze and predict the power grid behavior from data collected by IoT sensors. In PV systems, alongside other goals, ML has been explored to forecast their generation capacity. Accurate generation predictions make power grids more reliable amid fluctuations in demand and capacity, avoid power outages, prevent plant managers from penalties, and save costs [3]. More specifically, deep learning (DL) models have been applied to forecast PV power generation with encouraging results [4,5]. Although the use of these forecasting models contributes to more active, flexible, and intelligent smart grids [6], they may be vulnerable to adversarial examples. In these attacks, adversaries add maliciously crafted noise to legitimate input samples, driving the DL model to make wrong predictions [7]. This fact draws attention to the physical and cyber security of this kind of facility [8], especially when considering that industry practitioners are not equipped with measures to protect, detect, and respond to attacks on their ML models [9].

Different schemes [10–12] have been proposed recently to defend ML algorithms against adversarial examples. Studies [10,11] made use of adversarial training. In this technique, data used for model training include adversarial samples especially crafted to make it more resilient against this kind of attack. Conversely, Abdu-Aguye et al. [12] proposed an approach that detects adversarial samples during the test phase. Despite their encouraging results, these studies only focused on protecting ML models designed for classification tasks. The literature still lacks defense schemes for regression models, which can also be deeply affected by these attacks. In PV systems, these attacks represent a severe threat. An adversarial sample might make the forecasting model predict a much higher or lower generation capacity than the correct one. As operators use these predictions to coordinate multiple power plants that operate together to meet the energy demand, a high prediction error will eventually lead to wrong decisions, which can cause large-scale failures [13].

This work proposes a novel scheme to detect and mitigate adversarial samples inputted into DL regression models that forecast PV power generation. First, the approach extracts multiple features from the inputs forwarded to the forecasting system. These inputs are observations about the power plant generation capacity over time. The extracted features range from basic statistics such as minimum, maximum, and mean to spectral measures such as Hurst exponents. Their objective is to make a time series profile that allows for distinguishing natural observations from maliciously crafted ones. Then, a one-class classifier is employed to classify the feature vector as legitimate or malicious. If malicious behavior is detected, the observations are replaced by the last set of observations classified as legitimate. This means that the approach mitigates the attack, preventing the adversarial samples from reaching the forecasting system. The results showed that the proposed scheme could detect most of the adversarial samples and reduce significantly the error increase caused by the attacks.

The main contributions of this paper are:

- A novel scheme able to detect and mitigate adversarial attacks on DL regression models;
- A study of adversarial attacks on photovoltaic power generation forecasting;
- A comparison of Long-Short Term Memory (LSTM) and Temporal Convolutional Network (TCN) when under attack;
- An investigation regarding One-class Support Vector Machine (OCSVM) and Local Outlier Factor (LOF) as detectors of adversarial examples.

The remaining of this paper is organised as follows: Section 2 presents the background about time series and adversarial ML along with the related work. In Section 3, the proposed approach is discussed. Section 4 shows the materials and methods used during the proposal's evaluation, while Section 5 discusses the results. Finally, Section 6 draws the final conclusions.

2. Background

2.1. Time Series

A time series is a data sequence in a particular period. These data can produce different values at distinct moments in time. Formally, it can be defined as an ordered set of observations $X = [x_1, x_2, \dots x_T]$ in which T corresponds to the length of the series [14]. The forecasting task consists of finding a function f that predicts the h-th future value in any time t, i.e., $x_t + h$ based on i past values:

$$\hat{x}_{t+h} = f(x_{t-(i-1)}, x_{t-(i-2)}, \dots, x_{t-1}, x_t), \quad i \le t, \tag{1}$$

where i represents the input window size and h, the forecast horizon. When the latter is equal to one, the forecasting task is referred to as a one-step-ahead forecast. Otherwise, it is known as a multi-step ahead forecast. In a supervised training of f, t must also attend the condition $t \leq T - h$. Moreover, time series can present seasonality, which occurs when regular patterns are captured in the series. Seasonal events are phenomena that occur, for instance, daily at a certain time, every day, or in a certain month every year.

2.2. *Adversarial Machine Learning*

Solutions that rely on ML might suffer attacks based on adversarial examples [7]. Adversarial inputs are very similar to benign ones but tailored to maximize the model's prediction error. Three aspects of these attacks are worthy of discussing in this section: their classification, adversarial example generation, and defense strategies.

2.2.1. Attack Classification

An attack may be classified according to its specificity. In targeted attacks, the attacker focuses on specific system instances (e.g., specific users, periods, or inputs). Conversely, in untargeted attacks, the attacker aims at any instance of indiscriminate attacks.

Adversarial examples can be used at the training or test phases of the ML pipeline. A poisoning attack, also known as causative, occurs when the attacker can access and modify training data. Data access attacks are also related to the training phase but are more restricted. In these attacks, the attacker can access but not modify the training data. They may then use the retrieved data to induce substitute learning models useful for attacks in the test phase. An exploratory attack occurs when the attacker can modify only the test data [15].

The attacker's knowledge is another relevant feature, which might differ according to the level of access to the system components: training data, feature space, and learning algorithm. The latter may also involve the knowledge of the loss function and the trained hyper-parameters. The attacker's knowledge can be classified according to the access to these three components [16]:

- A white-box attack, which implies that the attacker has access to the entire set of components.
- A black-box attack, which implies that the attacker lacks substantial knowledge about the system components.
- A gray-box attack, which lies between the previous attacks. In this case, the attacker may have partial access to the training data, knowing the training algorithm or the feature space.

When the attacker lacks knowledge about the learning algorithm, an alternative is defining a surrogate/substitute model. This leads to the concept of transferability, which means that adversarial examples designed for a specific model can also affect another model [17].

2.2.2. Adversarial Examples Generation

The attacker generates an adversarial input to fool the ML model based on their knowledge about the target. By accessing training data or gathering information about the model, the attacker can make inputs that look like the legitimate ones but carry a perturbation specially crafted to explore the model's vulnerabilities [18]. Most of the methods for crafting adversarial examples were originally designed for images. A Fast Gradient Sign Method (FGSM) [19] is one of the most notable methods, being also the basis for later methods [20,21]. In an FGSM attack, the perturbation η is given by Equation (2):

$$\eta = \epsilon * sign(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y)), \tag{2}$$

where ϵ corresponds to the coefficient that controls the perturbation magnitude, $\mathbf{x}$ to the input to the model, y to the output associated with $\mathbf{x}$, θ to the weights of the adversarial model, and $J(.)$ to the loss function. The malicious sample to be inputted to the target results from adding η and $\mathbf{x}$. FGSM is computationally cheap since it only needs the gradient sign, which can be quickly obtained. Although it was designed to compute adversarial image perturbations, Santana et al. [13] showed that FGSM is also effective at making adversarial examples to degrade the prediction performance of DL models in PV power generation forecasting.

2.2.3. Defense Approaches

In the image processing literature, defense approaches include network distillation [22], adversarial retraining [23], randomisation [24], denoising [25], and adversarial example detection during test time [26]. The main idea behind adversarial example detection relies on training a classifier to detect adversarial inputs, distinguishing them from legitimate ones.

ML algorithms might be used for this task. They have been successful in detecting attacks in traditional computer systems and are also promising to detect attacks in smart grids and in time series [27–29]. In this kind of proposal, data about the target's behavior are gathered and then used to feed a ML-based classifier, which learns to classify the behavior as malicious or legitimate according to its characteristics [30,31]. As the types of attack change so does the type of analyzed data. For example, to detect network-based attacks, variables related to network traffic such as the packets per second rate or average packet size are investigated. False data injection in smart grids, on the other hand, can be detected through the analysis of measurements about the power grid state such as current flow and voltage magnitude. In short, these proposals gather data that are sensitive to an attack and use them to distinguish legitimate from malicious instances. The same rationale may be successfully applied to detect adversarial examples in the domain of PV generation.

Addressing the topic of adversarial examples in smart grids and time series, Chen et al. [32] evaluated the adversarial examples impact on feed-forward Neural Network (NN) and Recurrent Neural Network (RNN) models for simulated data on power quality classification and load forecasting, respectively. Based on the results, the authors encouraged more discussion towards increasing the robustness of models implemented in power systems.

Fawaz et al. [7] adapted FGSM and Basic Iterative Method to univariate time series classification and performed attacks against DL models. These attacks achieved an average reduction in the model's accuracy of 43.2% and 56.89%, respectively, and the experiments showed that FGSM allows real-time adversarial sample generation. The authors claim that their work is the first to consider the vulnerability of DL models concerning time series examples.

Niazazari and Livani [10] performed attacks on a multiclass Convolutional Neural Network (CNN) trained on simulated data. The targeted model classifies power grid events such as line energization, capacitor bank energization, or fault. The attacks were generated using FGSM and Jacobian-based Saliency Map Attack (JSMA) algorithms and showed significant potential to make the CNN-based model misclassify the tampered input.

Karim et al. [11] proposed using an adversarial transformation network to attack 1-Nearest Neighbor Dynamic Time Warping (1-NN DTW) and Fully Convolutional Network (FCN) models, trained on 42 classification datasets, showing their susceptibility to adversaries. They used the retraining defense strategy to improve the models' robustness. Abdu-Aguye et al. [12] proposed using OCSVM to classify samples as original or perturbed. The work was based on the attacks and datasets presented in [7]. The authors claimed to reach 90% detection accuracy on most datasets and up to 97% in the best case.

Table 1 summarizes the comparisons among the reviewed studies. This work addresses an important limitation found in the reviewed literature: the lack of protection for regression models. In other words, most researchers in adversarial ML are devoted to tackling classification focused on image processing tasks. As observed in a previous work [13], adversarial examples can also affect regression models, which are usually the core of PV generation forecasting. Among the related works, only Chen et al. [32] addressed this possibility, but they did not propose a defense solution against these attacks. All other proposals are aimed at attacks against classification models.

Table 1. Comparison among the related works. C stands for classification and R for regression.

Reference	Dataset	Model	Output	Attack	Transferability	Defence
Chen et al. (2018)	Simulated data on power quality and building load	NN and RNN	C and R	FGSM	Yes	-
Favaz et al. (2019)	Several univariate time series	ResNet	C	FGSM and BIM	No	-
Niazazari and Livani (2020)	Simulated data of grid events	CNN	C	FGSM and JSMA	No	Adversarial training
Karim et al. (2020)	Several time series datasets	1-NN DTW and FCN	C	FGSM and ATN	No	Adversarial training
Abdu-Aguye et al. (2020)	Several univariate time series	ResNet	C	FGSM and BIM	No	OC-SVM

3. Proposed Approach

Attacks involving adversarial examples against forecasting models consist of multiple steps. They begin with the attacker exploring any vulnerability that allows for accessing training data. Using these data, the attacker induces a model to craft malicious perturbations. Then, the attacker needs to find a breach to tamper with the data inputted into the forecasting model. After achieving this goal, the attacker can add malicious perturbations into the input data and complete the attack. As this attack requires breaking into multiple systems through various steps, multiple defense mechanisms are needed to tackle it. Detecting malicious inputs and preventing forecasting models from processing them may provide protection when other defense lines have already been violated.

This work proposes an approach based on one-class classifiers to detect and replace malicious inputs over power generation data in a PV plant. It assumes that adversarial examples can be distinguished from legitimate ones because they are intentional anomalies [33]. Even malicious inputs crafted to be as similar as possible to legitimate ones might carry distinguishable characteristics. Figure 1 provides an approach's overview.

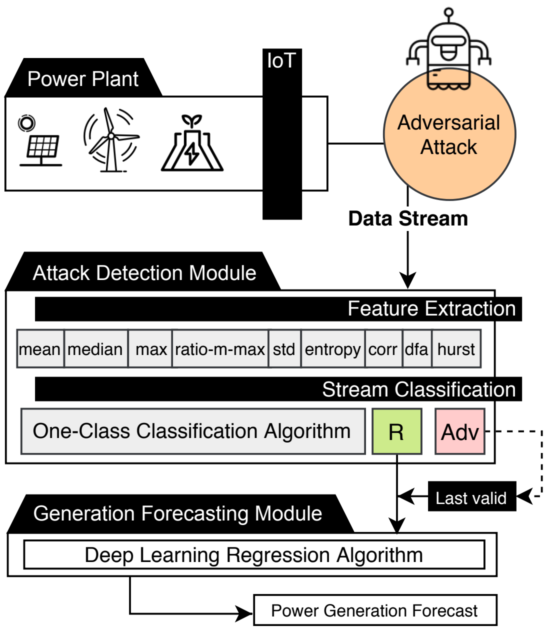

Figure 1. Proposed approach overview.

When new data instances from the power plant are forwarded to the Generation Forecasting Module, they are first assessed by the Attack Detection Module. This module organizes the data instances in windows of length i, which is the input window length of the forecasting model, as explained in Section 2.1. Then, the Attack Detection Module extracts the following features from each window: Minimum, Mean, Median, Maximum, Standard deviation, Ratio between Mean and Maximum, Ratio between Minimum and Maximum, Entropy, Correlation, Detrended fluctuation analysis (DFA), and Hurst Exponent. They make up a statistical profile of each window, which is intended to evince the differences between legitimate and maliciously crafted data.

After being extracted, the feature vector feeds an ML-based detector, more specifically, a one-class classifier. This kind of ML model is usually employed for anomaly detection. The most important one-class classifier's characteristic is the need for samples from only one class to be trained. In this work, the one-class classifier is trained using only legitimate data. As it might be hard to find samples from malicious data, this aspect of one-class algorithms is particularly useful for the proposed approach.

The one-class classification model then analyzes the feature vector extracted from the input window and classifies it as legitimate or malicious. When a malicious input is detected, the window is replaced by the most recent window classified as legitimate. Therefore, it prevents malicious data from being forwarded to the Generation Forecasting Module, while ensuring that the forecasting process keeps receiving inputs. Finally, the Generation Forecasting Module employs a DL model to make the predictions.

4. Materials and Methods

4.1. Dataset

The power generation samples are obtained from a PV plant that started to operate in November 2019 at the State University of Londrina campus (Brazil). This power plant is a typical IoT system that contains sensors connected to the Internet through a wireless network and transmits data to be processed in the cloud. More specifically, the plant has 1020 solar panels and sensors that collect observations about solar power generation every 15 min. Thus, 96 observations about the plant performance are collected each day. The plant's generation capacity is 489.6 MWh/year. Some variations in the collected samples may occur primarily due to two factors. Firstly, they depend on the weather condition. Rainy or cloudy days show a considerable disparity in sample values collected on sunny days. Secondly, the quality of the collection is also subject to interference from dirt that can accumulate on the solar panels, such as leaves from trees that surround the PV plant. These variations are also meaningful to calibrate the forecasting models.

All collected data are transmitted online to a private cloud maintained by the plant vendor, where the data are stored and can be accessed for operation and control. For the training of forecasting models, data from December 2019 to June 2020 was chosen. Moreover, 20% of the training data was used for hyper-parameter tuning of each model. For testing, data from July and August were employed.

4.2. Threat Model

Our threat model is based on targeted gray-box attacks that use FGSM for crafting adversarial examples. A targeted attack means that not all inputs are maliciously manipulated. The attacker picks specific inputs or targeted instances to manipulate according to some criteria. Modeling the behavior of attackers is an intricate task and exhaustive options are possible. For practical purposes, three different patterns were defined to select attacked instances:

(1) Random: the attacker picks the targeted instances at random. In this pattern, the attack can be confused with the plant intrinsic noise;

(2) Intermittent (inter): every targeted instance is followed by a non-attacked instance and vice versa. In [34], this pattern showed to be hardly detected by an estimation-based detector;

(3) Sinusoidal (sin): a group of targeted instances is followed by a group of non-attacked instances and vice versa. This function takes as an argument the instance index in radians. If the result is negative or zero, the instance is attacked. This pattern corresponds to a smoother variation of the intermittent pattern.

Figure 2 depicts the attack patterns.

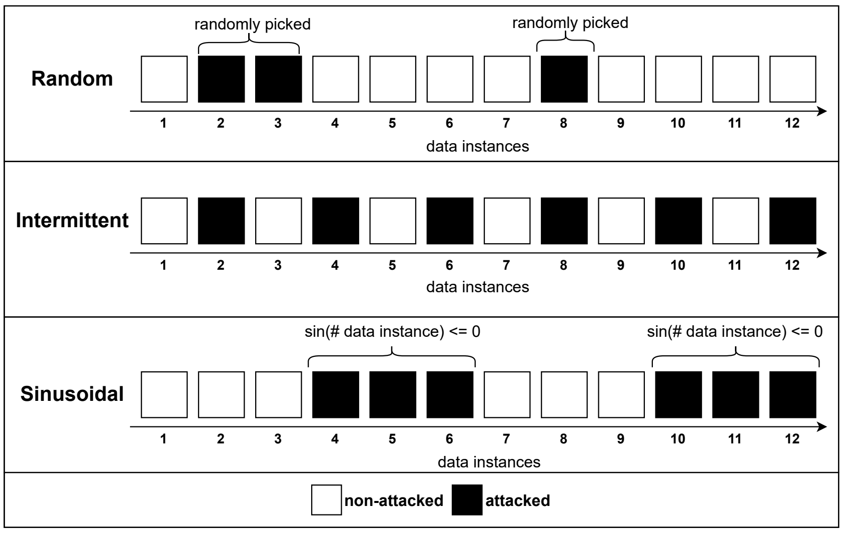

Figure 2. Attack patterns' (Random, Intermittent, and Sinusoidal) representations.

To understand how gray-box FGSM attacks can be launched against a forecasting model, it is necessary to recall first that the models addressed in this work have a training and a test phase. During the training phase, they use the training data to induce a regression model F. Then, during the test phase, they make predictions by using historical data inputted to F. In gray-box attacks [15], the attacker has limited knowledge about the target. Following this idea, it is assumed that the attacker can access a significant portion of the training data but has no knowledge about the model F induced by the target. Moreover, the attacker cannot modify the training data but can tamper with inputs during the test phase.

To overcome the lack of knowledge about F, the attacker induces a substitute model F', exploring the cross-technique transferability. This means that the attacker can analyze the attack circumstances and choose an algorithm that better fits their need to induce F'. In this work's scenario, an attacker could install malware or plug a rogue device at different points, ranging from the PV power plant to the cloud-based servers. If the algorithm behind the attack is a big consumer of CPU, memory, disk, or network resources, the defense systems that monitor these parameters can detect it. In this sense, employing a lightweight solution is an attacker's strategy to stay unnoticed. A costly ML algorithm can also make the requirements to run the attack very strict, hindering its execution. For being simpler than DL, successful in other adversarial scenarios [17] and still differentiable, logistic regression (LR) was adopted to build the substitute model.

Based on this strategy, the attacker uses the first half of their training data to build F'. Then, to compute the perturbation η in Equation (2), the attacker uses the second half of the training data as x and y along with the F' model. After calculating η, the attacker is ready to manipulate inputs and generate adversarial examples in the forecasting model's test phase. For a given legitimate input x_{test} that is forwarded to the forecasting model, the attacker will make an adversarial sample x'_{test} according to Equation (3):

$$x'_{test} = x_{test} + \eta \tag{3}$$

This adversarial example x'_{test} is inputted to the forecasting model instead of x_{test}, increasing F's prediction error. During the experiments, the ϵ value in Equation (2) was varied from 0.05 to 2 with steps of 0.05. In FGSM attacks, ϵ determines the attack magnitude.

4.3. Attack Detection Module

As presented in Section 3, 11 statistical features were extracted for each input window: Minimum, Mean, Median, Maximum, Standard deviation, Ratio between Mean and Maximum, Ratio between Minimum and Maximum, Entropy, Correlation, DFA, and Hurst Exponent. Entropy, Correlation, DFA, and Hurst were obtained using ndols [35] (https://nolds.readthedocs.io/en/latest/nolds.html#algorithms (accessed on 1 September 2021)) Python module.

As for the one-class classifier in the Attack Detection Module's core, OCSVM and LOF were explored. This kind of classifier has been successfully applied to fault detection in smart electric power systems [6]. OCSVM [36] creates hyperplanes (n-dimensional planes) that set boundaries around a region containing as much as possible of the training data. By doing so, OCSVM can identify whether an instance is within this area. LOF estimates a score, named Outlier Factor, which reflects the level of abnormality of each observation from a dataset [37]. It works based on the idea of local density. The k-Nearest Neighbors algorithm is applied to the data, and each data instance is given a locality, which is used to estimate the clusters' density.

During the experiments, the OCSVM hyper-parameter ν varied from 0.1 to 0.4 in 0.05 steps. As for LOF, the contamination alternated between 2.5×10^{-4} and 5×10^{-4}, and the number of neighbors varied from 20 to 45 in steps of 5.

4.4. Generation Forecasting Module

Studies related to data analysis in time series have been carried out for a long time [38,39]. ML-based applications have become more popular due to their high performance on data inference, outperforming even classical statistical models [40]. More specifically, DL techniques have played a fundamental role in reducing the regression approaches' error. This work explores TCN and LSTM, both DL models, to make predictions in time series.

LSTM is a type of RNN. Unlike some traditional neural networks, LSTM can remember the most useful information. This is possible thanks to its architecture. The networks that comprise the LSTM are connected in the form of loops. This process allows information to persist on the network. It also has a gating mechanism for learning long-term dependencies without losing short-term capability [41]. In particular, this neural network has been achieving important contributions in photovoltaic power generation forecasting [42].

TCN is a special type of CNN capable of handling a large amount of information. This processing is done through causal convolutions, which ensures that the model cannot violate the order in which the data are processed. TCN uses a one-dimensional fully-convolutional network architecture, where each hidden layer has the same length as the input layer, and zero padding of length. The hyper-parameters used for each of these models are shown in Table 2.

Table 2. Hyper-parameters experimented for tuning.

	Parameters	**Experimented Hyper-Parameters**
LSTM	Number of stacked layers	1, 2, 3
	Units	32, 64, 128
TCN	Number of filters	32, 64
	Kernel	2, 3
	Dilations	[1, 4, 12, 48], [1, 2, 4, 8, 12, 24, 48], [1, 4, 16, 32], [1, 2, 4, 8, 16, 32], [1, 3, 6, 12, 24], [1, 2, 6, 12, 24], [1, 2, 4, 8, 16], [1, 4, 16], [1, 2, 4, 8], [1, 4, 8]
	Blocks	1, 2

4.5. Evaluation Metrics

To compute the forecasting model performance, the Root Mean Squared Error (RMSE) was assessed for the test sets. This error metric tends to be more robust with undesirable large deviations [43]. F1-score was calculated to evaluate the detector. This metric describes the relation between two other metrics for classifiers, recall and precision. Precision measures the percentage of classified adversarial examples that are truly malicious. Recall consists of the effectiveness of the approach in identifying adversarial examples.

5. Results and Discussion

5.1. Adversarial Examples' Mitigation

This section assesses whether the Attack Detection Module effectively reduces the adversarial examples' impact over the prediction error. Alongside the detection mechanism, the mitigation approach is evaluated here. The mitigation function blocks samples classified as malicious and, at the same time, has to be able to replace them with samples that keep the prediction error low.

The tests were carried out as follows. First, the Generation Forecasting Module was executed to make predictions under non-attack and attack scenarios without the Attack Detection Module's aid. The same data used for the tests in Section 5.2 were employed here. The results show that LSTM had a better performance in terms of RMSE. LSTM obtained the lowest error in several scenarios: without attack and for ϵ with values of 0.05, 0.15, and 0.2. TCN outperformed LSTM just for $\epsilon = 0.1$. The second part of the tests reintroduced the Attack Detection Module in the pipeline. A remarkable reduction of RMSE for both models (LSTM and TCN) was observed using OCSVM or LOF at the Attack Detection Module's core. Figure 3 presents the results for all these scenarios.

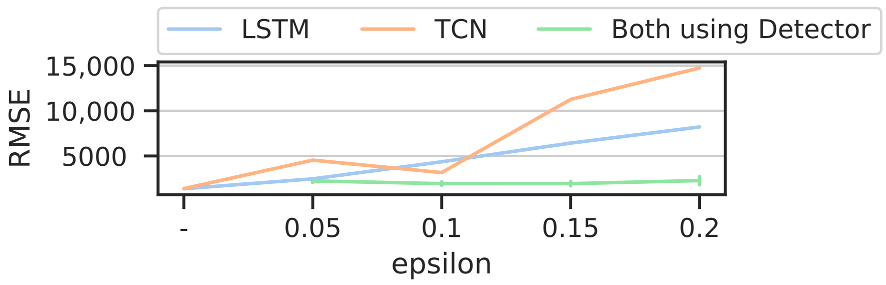

Figure 3. RMSE obtained using LSTM and TCN classifiers with and without detectors.

Table 3 presents RMSE obtained with all attack patterns and grouped by ϵ. TCN outperformed LSTM in all scenarios where the Attack Detection Module was present in the pipeline. This result suggests that TCN benefits more from the mitigation scheme than LSTM. The fact that LSTM is solidly grounded on the time series's sequential information can explain this outcome. As the mitigation scheme uses the most recent legitimate input, when the current input is malicious, the time series' sequence is eventually broken. TCN, which uses local and global information of the time series, handles this characteristic of the mitigation strategy better.

It is noteworthy that the error increase for the scenario with $\epsilon = 0.15$ was substantially reduced when the Attack Detection Module was used. In this scenario, the Attack Detection Module based on LOF reduced the increase in TCN's prediction error caused by adversarial examples from 711.21% to 19.70%.

Table 3. Comparison among RMSE obtained with different forecasting models and detectors. The lowest RMSE for each combination of epsilon/detector was highlighted in bold.

ϵ	Detector	Forecast RMSE * (% **)	
		LSTM	TCN
-	-	**1370.35**	1388.24
0.05	-	**2466.18** (79.97%)	4538.75 (226.94%)
0.10	-	4356.53 (217.91%)	**3160.94** (127.69%)
0.15	-	**6427.32** (369.03%)	11,261.55 (711.21%)
0.20	-	**8213.34** (499.36%)	14,746.09 (962.21%)
0.05	OCSVM	2433.76 (77.60%)	**2087.62** (50.38%)
0.10	OCSVM	2143.56 (56.42%)	**1764.01**(27.07%)
0.15	OCSVM	2198.36 (60.42%)	**1729.19** (24.56%)
0.20	OCSVM	2921.47 (113.19%)	**2358.44**(69.89%)
0.05	LOF	2427.25 (77.13%)	**2008.58** (44.68%)
0.10	LOF	2116.04 (54.42%)	**1719.30** (23.85%)
0.15	LOF	2161.84 (57.76%)	**1661.70** (19.70%)
0.20	LOF	2145.56 (56.57%)	**1686.87** (21.51%)

* Computed in W; ** RMSE increase between the current one and the corresponding RMSE; without attack.

5.2. Attack Detection Module's Efficacy in Detecting Adversarial Examples

OCSVM and LOF were applied as detectors using different hyper-parameters to find the most suitable classifier for the Attack Detection Module. Figure 4 shows box plots for LOF F1-Scores obtained by varying the Number of Neighbors (20, 25, 30, 35, 40, and 45) and Contamination (2.5×10^{-4} and 5×10^{-4}) over different attack patterns (random, intermittent, and sinusoidal) and ϵ values (0.05 to 2 in 0.05 steps). In box plots, boxes depict the range between the upper and lower quartiles, while horizontal lines inside the boxes represent the median. Vertical lines extending from the boxes illustrate the variability outside the quartiles. Individual points represent outliers.

The results for different numbers of neighbors show that lower values for this hyper-parameter deliver better results. The best average F1-score found in these tests was 86.05%, obtained with 20 neighbors. In contrast, the contamination hyper-parameter tests pointed out that the best average performances were obtained with the highest value for this hyper-parameter. With Contamination = 5×10^{-4}, the detector reached an average F1-Score of 87.94%. The standout LOF outcome was obtained by combining 40 neighbors and contamination = 5×10^{-4}, which resulted in an F1-Score of 95.86% for detecting adversarial examples following a sinusoidal pattern.

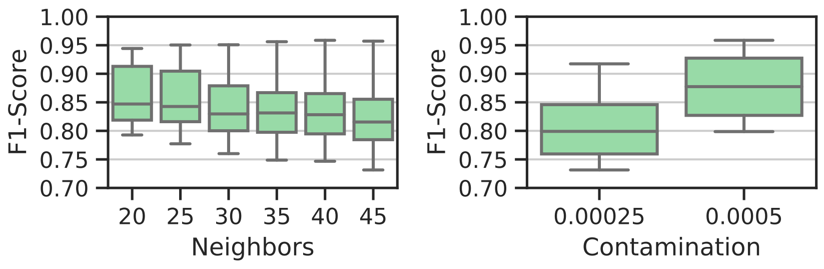

Figure 4. F1-Score obtained with LOF across several Number of Neighbors and Contamination hyper-parameters.

The tests for OCSVM with different values for ν (0.1, 0.15, 0.20, 0.25, 0.30, 0.35, and 0.40) over different attack patterns and ϵ showed that the proposed system can reach a better average F1-Score with lower values for this hyper-parameter, as presented in the box plots in Figure 5. For attacks based on the sinusoidal pattern and $\nu = 0.1$, the detector reached its best performance: F1-score = 93.12%.

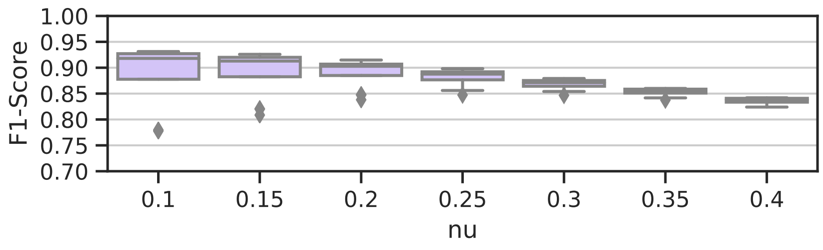

Figure 5. F1-Score from OCSVM using seven ν values.

To check whether there is a statistically significant difference between the performance of both classifiers, LOF and OCSVM, the Friedman's statistical test and the post-hoc test of Nemenyi were used. In this evaluation, three metrics were compared: F1-Score, precision, and recall. The Critical Difference (CD) demonstrates that the difference between two algorithms is significant if the gap between their ranks is larger than CD. Otherwise, no significant differences are found between them. Diagrams for these three metrics are presented in Figures 6–8. The metrics were collected considering all attack patterns and ϵ values, and the tests had a significance level of 95%. According to the statistical tests, there was a statistically significant difference between both models since the CD is equal to 0.57 and the distance between them is equal to 1.

The CD value equal to 0.57 is the same for all scenarios as the number of experiments and algorithms used are also the same. Consequently, there is statistical difference between the metrics evaluating the two models with the same value in all cases.

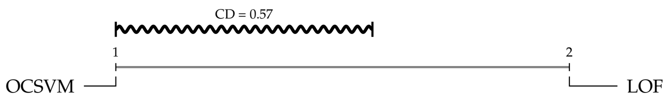

Figure 6. Critical Difference chart of F1-Score performance between LOF and OCSVM across all attack patterns and magnitudes (ϵ). It is possible to observe that there is no link between the first and second techniques, which demonstrates significant difference between OCSVM and LOF, with OCSVM as the superior one.

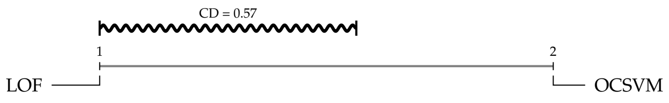

Figure 7. Critical Difference chart of Recall performance between LOF and OCSVM across all attack patterns and magnitudes (ϵ). It is possible to observe that there is no link between the first and second techniques, which demonstrates a significant difference between OCSVM and LOF, with LOF as the superior one.

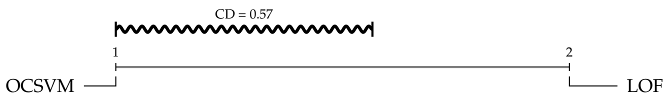

Figure 8. Critical Difference chart of Precision performance between LOF and OCSVM across all attack patterns and magnitudes (ϵ). It is possible to observe that there is no link between the first and second techniques, which demonstrates some significant difference between OCSVM and LOF, with OCSVM as the superior one.

The detector efficacy focusing on the influence of ϵ values and attack patterns was also analyzed. The experiments showed that LOF reached higher F1-score for lower ϵ, while OCSVM outperformed LOF for higher ϵ values, as Figure 9 shows. Clear differences in detection performance were not observed for each attack pattern. Figure 10 presents a box plot that depicts F1-Score results obtained with LOF and OCSVM considering the three attack patterns (random, inter, and sin). OCSVM achieved a higher median F1-Score than

LOF for the three attack patterns. Actually, the OCSVM median was very close to the third quartile of LOF and the minimum values of OCSVM were very close to the LOF median.

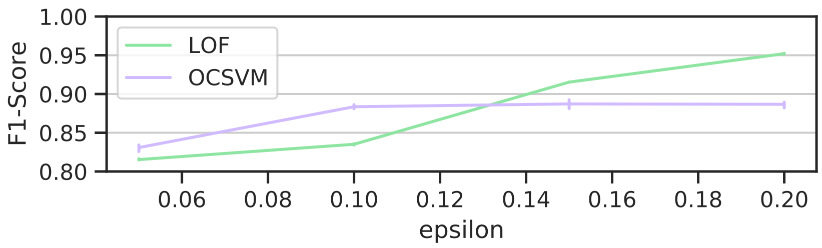

Figure 9. F1-Score variation from ϵ using LOF and OCSVM.

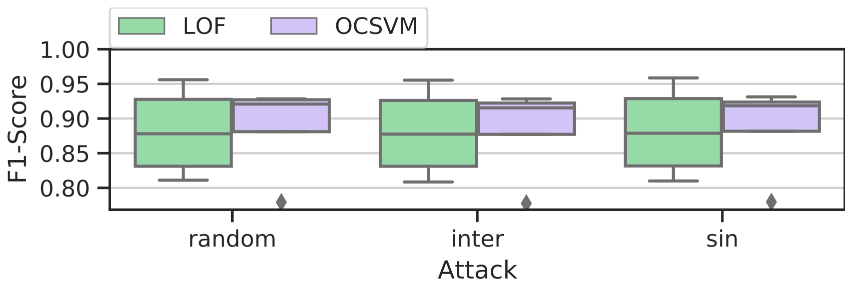

Figure 10. F1-Score variation of LOF and OCSVM grouped by attack pattern.

Lastly, the influence of the detection model, attack magnitude (ϵ), and attack pattern on the Attack Detection Module's efficacy was investigated. The Pearson correlation coefficient was employed to identify a linear relationship between each factor and the detection performance. A coefficient value of 0 means no correlation. On the other hand, a value close to -1 or 1 represents the full correlation. The obtained correlations were 0.016, 0.244, and 0.651, for attack pattern, detection model, and attack magnitude, respectively. This result suggests that the detection performance is more affected by the attack magnitude, while the attack pattern and the detection model have a low correlation to the detector efficacy.

5.3. Feature Importance

Seeking to provide more insights into what distinguishes FGSM adversarial samples from legitimate ones, the importance of each feature inputted to the Attack Detection Module was analyzed. Spectral Feature Selection for Supervised and Unsupervised Learning (SPEC) [44] was used to this end. Figure 11 shows the features sorted by their importance. Hurst exponent (hurst), median, entropy, ratio between mean and maximum (ratio-mean-max), and correlation (corr) were the most promising ones with roughly the same importance. Mean, standard deviation (std), and maximum (max) showed slightly worse performance than the best ones. Despite its high importance, DFA is clearly less important than the other ones. In short, the computed features' importance suggests that a great part of them contribute significantly to distinguish legitimate and adversarial samples, except the features related to the minimum value (the minimum itself and the ratio between minimum and maximum features).

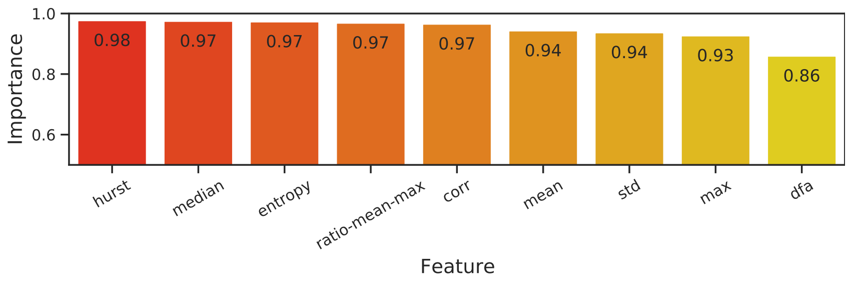

Figure 11. Feature importance computed using SPEC. The highest values reveal the most important features.

5.4. Discussion

Considering two different classifiers (LOF and OCSVM), three attack patterns (random, intermittent, and sinusoidal), and a broad range of attack magnitudes, the results showed that the proposed approach was consistently effective at detecting adversarial examples over several situations. The approach successfully detected low-magnitude attacks, which are particularly challenging due to their small difference to legitimate samples, and achieved excellent performance in detecting high-magnitude attacks. The variation of attack patterns did not affect the detection capacity. Moreover, OCSVM and LOF both had a good performance, but OCSVM was statistically superior to LOF. Abdu-Aguye et al. [12] also achieved high accuracy at detecting FGSM adversarial examples with an OCSVM-based scheme. Unlike this work, their scheme focused on defending classification models and did not vary attack patterns and attack magnitudes. Despite these methodological differences, the high efficacy reported by both studies suggests that one-class classifiers are a promising option to address this issue.

Almost all features extracted from the forecasting model input showed to be good indicators of artificial presence within the analyzed data. First, this suggests that adversarial examples affect the analyzed window's basic features, such as minimum, maximum, and median. Moreover, this result implies that features related to the time series spectral behavior, such as the Hurst exponent, are influenced by these artificial manipulations.

Combining the detection approach with a mitigation mechanism allowed a significant reduction in the error increase caused by adversarial samples. For high-magnitude attacks, the error increase plummeted from figures above 700% to roughly 20%. The results were also relevant for low-magnitude attacks, dropping from above 200% to around 45% in the worst case. Both TCN and LSTM could benefit from using the attack detection and mitigation mechanism, but slightly better results were found for TCN. Other studies [10,11] that followed a different mitigation strategy (e.g., adversarial training) also reported a positive impact on the target model robustness towards adversarial examples. Nevertheless, with a simple mitigation scheme backed by an effective detection approach, this work achieved a positive outcome for attack mitigation without requiring adversarial examples during the training phase.

6. Conclusions

DL models are great options to forecast the generation capacity of PV power plant, but they are vulnerable to adversarial examples: as the results showed, the forecasting error under attack increased up to 962.21% when compared to the forecasting error in non-attacked conditions.

On the other hand, detecting and discarding these examples reduces the damage to the forecasting model accuracy: in the worst case, the error increased 77.60% when compared to the forecasting error in non-attacked conditions. In this sense, schemes that detect adversarial examples and mitigate them should not be neglected to avoid the malfunction of the power plant.

Future work includes investigating other methods to defend regression models against adversarial examples, testing the proposed scheme over different attack methods and domains. Furthermore, the proposed mitigation approach will be extended as it is possibly a point that can be changed to reduce the error increase caused by attacks even more.

Author Contributions: Conceptualization, E.J.S., B.B.Z., and S.B.J.; methodology, E.J.S. and S.B.J.; investigation, E.J.S. and R.P.S.; writing—original draft preparation, E.J.S.; writing—review and editing, R.P.S., B.B.Z., and S.B.J.; visualization, E.J.S. and S.B.J.; supervision, S.B.J. All authors have read and agreed to the published version of the manuscript.

Funding: Coordenação de Aperfeicoamento de Pessoal de Nível Superior: Finance code 001; National Council for Scientific and Technological Development: 420562/2018-4; National Council for Scientific and Technological Development: 440045/2020-7; National Council for Scientific and Technological Development: 309863/2020-1; Amazon (United States): CNPq/AWS 032/2019.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Restrictions apply to the availability of these data. Data were obtained from the administration of the State University of Londrina and are available from the authors with the permission of the university.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ARIMA	Autoregressive Integrated Moving Average
BIM	Basic Iterative Method
CD	Critical Difference
CNN	Convolutional Neural Network
DL	Deep Learning
DFA	Detrended Fluctuation Analysis
DTW	Dynamic Time Warping
FCN	Fully Convolutional Network
FGSM	Fast Gradient Sign Method
JSMA	Jacobian-based Saliency Map Attack
LOF	Local Outlier Factor
LR	Logistic Regression
LSTM	Long Short-term Memory Network
ML	Machine Learning
OCSVM	One Class Support Vector Machine
PV	Photovoltaic
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Networks
SPEC	Spectral Feature Selection for Supervised and Unsupervised Learning
TCN	Temporal Convolutional Neural Network

References

1. Das, U.K.; Tey, K.S.; Seyedmahmoudian, M.; Mekhilef, S.; Idris, M.Y.I.; Van Deventer, W.; Horan, B.; Stojcevski, A. Forecasting of photovoltaic power generation and model optimization: A review. *Renew. Sustain. Energy Rev.* **2018**, *81*, 912–928. [CrossRef]
2. López-Vargas, A.; Fuentes, M.; Vivar, M. IoT Application for Real-Time Monitoring of Solar Home Systems Based on Arduino™ With 3G Connectivity. *IEEE Sens. J.* **2019**, *19*, 679–691. [CrossRef]
3. Antonanzas, J.; Osorio, N.; Escobar, R.; Urraca, R.; Martinez-de Pison, F.J.; Antonanzas-Torres, F. Review of photovoltaic power forecasting. *Sol. Energy* **2016**, *136*, 78–111. [CrossRef]
4. Wang, K.; Qi, X.; Liu, H. A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network. *Appl. Energy* **2019**, *251*, 113315. [CrossRef]
5. Yen, C.F.; Hsieh, H.Y.; Su, K.W.; Leu, J.S. Predicting Solar Performance Ratio Based on Encoder-Decoder Neural Network Model. In Proceedings of the 2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), Dublin, Ireland, 28–30 October 2019; pp. 1–4.

6. Ibrahim, M.S.; Dong, W.; Yang, Q. Machine learning driven smart electric power systems: Current trends and new perspectives. *Appl. Energy* **2020**, *272*, 115237. [CrossRef]
7. Fawaz, H.I.; Forestier, G.; Weber, J.; Idoumghar, L.; Muller, P.A. Adversarial attacks on deep neural networks for time series classification. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8.
8. Hua, B.; Zhang, K.; Wei, H.; Zhang, J.; Xie, L. A Multilevel Fire Detection Platform Based on Multi-Source Heterogeneous Information Fusion. In Proceedings of the 2020 International Conference on Pattern Recognition and Intelligent Systems, Athens, Greece, 30 July–2 August 2020; pp. 1–5.
9. Kumar, R.S.S.; Nyström, M.; Lambert, J.; Marshall, A.; Goertzel, M.; Comissoneru, A.; Swann, M.; Xia, S. Adversarial machine learning-industry perspectives. In Proceedings of the 2020 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 21–21 May 2020; pp. 69–75.
10. Niazazari, I.; Livani, H. Attack on Grid Event Cause Analysis: An Adversarial Machine Learning Approach. In Proceedings of the 2020 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), Washington, DC, USA, 17–20 February 2020; pp. 1–5.
11. Karim, F.; Majumdar, S.; Darabi, H. Adversarial attacks on time series. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 3309–3320. [CrossRef]
12. Abdu-Aguye, M.G.; Gomaa, W.; Makihara, Y.; Yagi, Y. Detecting Adversarial Attacks In Time-Series Data. In Proceedings of the ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 3092–3096.
13. Santana, E.J.; Silva, R.P.; Zarpelão, B.B.; Barbon Junior, S. Photovoltaic Generation Forecast: Model Training and Adversarial Attack Aspects. In Proceedings of the Brazilian Conference on Intelligent Systems, Rio Grande, Brazil, 20–23 October 2020; pp. 634–649.
14. Parzen, E. An approach to time series analysis. *Ann. Math. Stat.* **1961**, *32*, 951–989. [CrossRef]
15. Tabassi, E.; Burns, K.J.; Hadjimichael, M.; Molina-Markham, A.D.; Sexton, J.T. A Taxonomy and Terminology of Adversarial Machine Learning. *NIST IR* **2019**, 1–29. [CrossRef]
16. Biggio, B.; Roli, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognit.* **2018**, *84*, 317–331. [CrossRef]
17. Papernot, N.; McDaniel, P.; Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv* **2016**, arXiv:1605.07277.
18. Yuan, X.; He, P.; Zhu, Q.; Li, X. Adversarial examples: Attacks and defenses for deep learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 2805–2824. [CrossRef] [PubMed]
19. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
20. Rozsa, A.; Rudd, E.M.; Boult, T.E. Adversarial diversity and hard positive generation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Las Vegas, NV, USA, 27–30 June 2016; pp. 25–32.
21. Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; Li, J. Boosting adversarial attacks with momentum. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9185–9193.
22. Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. In Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2016; pp. 582–597.
23. Tian, J.; Li, T.; Shang, F.; Cao, K.; Li, J.; Ozay, M. Adaptive Normalized Attacks for Learning Adversarial Attacks and Defenses in Power Systems. In Proceedings of the 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), Beijing, China, 21–23 October 2019; pp. 1–6.
24. Liu, X.; Cheng, M.; Zhang, H.; Hsieh, C.J. Towards robust neural networks via random self-ensemble. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 369–385.
25. Meng, D.; Chen, H. Magnet: A two-pronged defense against adversarial examples. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 15 September 2017; pp. 135–147.
26. Lu, J.; Issaranon, T.; Forsyth, D. Safetynet: Detecting and rejecting adversarial examples robustly. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 446–454.
27. Zarpelão, B.B.; Barbon, S.; Acarali, D.; Rajarajan, M. How Machine Learning Can Support Cyberattack Detection in Smart Grids. In *Artificial Intelligence Techniques for a Scalable Energy Transition*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 225–258.
28. Martins, N.; Cruz, J.M.; Cruz, T.; Abreu, P.H. Adversarial machine learning applied to intrusion and malware scenarios: A systematic review. *IEEE Access* **2020**, *8*, 35403–35419. [CrossRef]
29. Li, J.; Yang, Y.; Sun, J.S.; Tomsovic, K.; Qi, H. Conaml: Constrained adversarial machine learning for cyber-physical systems. In Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, Hong Kong, China, 7–11 June 2021; pp. 52–66.
30. Duy, P.T.; Khoa, N.H.; Nguyen, A.G.T.; Pham, V.H. DIGFuPAS: Deceive IDS with GAN and Function-Preserving on Adversarial Samples in SDN-enabled networks. *Comput. Secur.* **2021**, *109*, 102367. [CrossRef]
31. Anthi, E.; Williams, L.; Javed, A.; Burnap, P. Hardening machine learning denial of service (DoS) defenses against adversarial attacks in IoT smart home networks. *Comput. Secur.* **2021**, *108*, 102352. [CrossRef]

32. Chen, Y.; Tan, Y.; Deka, D. Is machine learning in power systems vulnerable? In Proceedings of the 2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), Aalborg, Denmark, 29–31 October 2018; pp. 1–6.
33. Bulusu, S.; Kailkhura, B.; Li, B.; Varshney, P.; Song, D. *Anomalous Instance Detection in Deep Learning: A Survey*; Technical Report; Lawrence Livermore National Lab.(LLNL): Livermore, CA, USA, 2020.
34. Kontouras, E.; Tzes, A.; Dritsas, L. Hybrid Detection of Intermittent Cyber-Attacks in Networked Power Systems. *Energies* **2019**, *12*, 4625. [CrossRef]
35. Schölzel, C. Nonlinear Measures for Dynamical Systems. 2019. Available online: https://zenodo.org/record/3814723#.YUzRj7hKjIU (accessed on 1 September 2021).
36. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
37. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; pp. 93–104.
38. Peng, W.; Liu, R.; Wang, R.; Cheng, T.; Wu, Z.; Cai, L.; Zhou, W. EnsembleFool: A method to generate adversarial examples based on model fusion strategy. *Comput. Secur.* **2021**, *107*, 102317. [CrossRef]
39. Appiah, B.; Qin, Z.; Abra, A.M.; Kanpogninge, A.J.A. Decision tree pairwise metric learning against adversarial attacks. *Comput. Secur.* **2021**, *106*, 102268. [CrossRef]
40. Cerqueira, V.; Torgo, L.; Soares, C. Machine Learning vs Statistical Methods for Time Series Forecasting: Size Matters. *arXiv* **2019**, arXiv:1909.13316.
41. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
42. Kim, D.; Kwon, D.; Park, L.; Kim, J.; Cho, S. Multiscale LSTM-Based Deep Learning for Very-Short-Term Photovoltaic Power Generation Forecasting in Smart City Energy Management. *IEEE Syst. J.* **2020**, *15*, 346–354. [CrossRef]
43. Ahmed, R.; Sreeram, V.; Mishra, Y.; Arif, M. A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization. *Renew. Sustain. Energy Rev.* **2020**, *124*, 109792. [CrossRef]
44. Zhao, Z.; Liu, H. Spectral feature selection for supervised and unsupervised learning. In Proceedings of the 24th International Conference on Machine Learning, Corvalis, OR, USA, 20–24 June 2007; pp. 1151–1157.