Universidade Estadual de Londrina

EVERTON JOSE SANTANA

PHOTOVOLTAIC GENERATION FORECAST IN ADVERSARIAL SETTINGS

Londrina
2021

EVERTON JOSE SANTANA

PHOTOVOLTAIC GENERATION FORECAST IN ADVERSARIAL SETTINGS

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Sylvio Barbon Jr.

Londrina
2021

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

S232 Santana, Everton Jose.
Photovoltaic Generation Forecast in Adversarial Settings / Everton Jose Santana. - Londrina, 2021.
76 f. : il.

Orientador: Sylvio Barbon Junior.
Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2021.
Inclui bibliografia.

1. Predição de Séries Temporais - Tese. 2. Aprendizado de Máquina em Ambientes com Adversários - Tese. 3. Aprendizado Profundo - Tese. 4. Energia Fotovoltaica - Tese. I. Barbon Junior, Sylvio. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU 519

EVERTON JOSE SANTANA

PHOTOVOLTAIC GENERATION FORECAST IN ADVERSARIAL SETTINGS

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

_____

Orientador: Prof. Dr. Sylvio Barbon Jr.
Universidade Estadual de Londrina - UEL

_____

Prof. Dr. Bruno Bogaz Zarpelão
Universidade Estadual de Londrina – UEL

_____

Prof. Dr. Rafael Gomes Mantovani
Universidade Tecnológica Federal do Paraná,
Campus Apucarana – UTFPR

_____

Prof. Dr. Luiz Fernando Carvalho
Universidade Tecnológica Federal do Paraná,
Campus Apucarana – UTFPR

Londrina, 18 de fevereiro de 2021.

This master's thesis is dedicated to those who desire and strive to build a better future.

ACKNOWLEDGEMENTS

I would like to start by thanking my family. Every sacrifice you have done to grant me the opportunity to follow my dreams will not be forgotten.

I wish to express my gratitude to my supervisor, professor Sylvio Barbon Jr., for having given the opportunity to an unknown electrical engineering undergraduate student to join his research group. Thanks for being so inspiring, dedicated and supportive in so many years of collaboration and friendship.

I would like to extend my thanks to professors Bruno, Rafael and Luiz for accepting being part of the evaluation process of this work and suggesting improvements that certainly improved this work. Thanks also for the teachings and collaborations.

My gratitude also for all REMID Laboratory friends and co-authors during this master's period. I learned a lot from you. A special thanks to Saulo Martiello Mastelini, for all encouragement and brotherhood, and Ricardo Petri Silva, for the contribution with the experiments of this work.

I would like to thank the financial support of Coordination for the Improvement of Higher Education Personnel (CAPES) - Finance Code 001 and Brazilian National Council for Scientific and Technological Development (CNPq)/Amazon Web Service (#440045/2020-7).

I kindly thank those who have been more intimately part of my life. I value so much sharing life with you and learning from our experience exchange. You were so important for me, especially in this pandemic period.

My thanks to the one who has the tiller in his hands. Thanks for every path you led me, the people I could meet in the way and for allowing to, at least minimally, enter into your mysteries.

"The deeper our insight into the mysteries of nature, the more we have to be able to stand up to the powers within ourselves."

Joseph Kentenich

SANTANA, E. J.. **Previsão de Geração Fotovoltaica em Ambientes com Adversários**. 2021. 76f. Dissertação (Mestrado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina, 2021.

RESUMO

A previsão de geração fotovoltaica, assim como outros cenários de séries temporais, é uma tarefa desafiadora. Em particular para esse tipo de geração, a dependência de fatores meteorológicos aumenta a dificuldade de obter métodos preditivos com desempenho adequado. A maioria das soluções atuais relacionadas à previsão de geração fotovoltaica é baseada em algoritmos de aprendizado de máquina, que normalmente oferecem desempenho superior aos métodos estatísticos tradicionais. Contudo, modelos de aprendizado de máquina, mais especificamente de aprendizado profundo, têm se mostrado vulneráveis a ataques adversariais ao longo de sua execução, aumentando o erro do modelo. Considerando que o aumento no erro preditivo em uma usina pode causar grandes danos, antecipar possíveis ataques e proteger-se deles é uma atitude essencial. Nesse sentido, três principais eixos são investigados: i) obter modelos preditivos de geração fotovoltaica satisfatórios, ii) analisar as vulnerabilidades desses modelos em relação a ataques adversariais e iii) defendê-los desse tipo de ataque. Primeiramente, quatro técnicas de séries temporais são exploradas para a predição de geração, denominadas *Naive* - referência para séries temporais -, modelo Autorregressivo Integrado de Médias Móveis - da área estatística -, e Redes de Memória de Curto e Longo Prazo e Rede Convolucional Temporal - ambas da família de aprendizado profundo. Essas técnicas foram utilizadas na predição de geração com 15 minutos e 24 horas de antecedência, tendo como entrada apenas dados históricos de geração. Neste ponto, foi analisada a influência do número de amostras de treino com atualizações mensais no desempenho dos modelos preditivos e qual deles forneceu menor erro. Posteriormente, foi investigado como os dados de séries temporais univariadas puderam ser modificados por um ataque adversarial com o intuito de degradar o desempenho do modelo por meio de transferência entre técnicas. Para isso, o algoritmo de ataque *fast gradient sign method* (FGSM), junto com um modelo substituto de regressão logística, foi usado para realizar ataques contra os modelos durante o período de teste. Por último, foi proposto um mecanismo para detectar ataques adversariais durante a operação da usina e defendê-la deles. A detecção foi baseada em atributos estatísticos e classificadores *Local Outlier Factor* e *One-Class Support Vector Machine*. Este trabalho foi baseado em dados reais de uma usina solar e os resultados mostram que os modelos de aprendizado profundo podem auxiliar no planejamento dessa usina, ao passo que mecanismos de defesa contra ataques adversariais devem ser adotados na previsão de geração fotovoltaica.

Palavras-chave: Predição de Séries Temporais. Aprendizado Profundo. Aprendizado de Máquina em Ambientes com Adversários. Rede Elétrica Inteligente. Energia Fotovoltaica. Geração de Energia.

SANTANA, E. J.. **Photovoltaic Generation Forecast in Adversarial Settings**. 2021. 76p. Master's Thesis (Master in Science in Computer Science) – State University of Londrina, Londrina, 2021.

ABSTRACT

Forecasting photovoltaic (PV) power generation, as in many other time series scenarios, is a challenging task. Especially for PV generation, the dependence on meteorological factors increases the difficulty to design satisfactory methods. Most current solutions for PV generation forecasting are grounded on machine learning (ML) algorithms, which usually outperform traditional statistical-based methods. However, solutions based on ML and, more specifically, deep learning (DL) have been found vulnerable to adversarial attacks throughout their execution, increasing the model error. Given that error increases in a PV plant can be very damaging, anticipating possible adversarial attacks and securing against them is essential. Bearing this in mind, three main segments are investigated in this work: i) obtaining satisfactory forecasting models for PV generation, ii) analyzing the model's vulnerability to adversarial attacks and iii) defending them against these attacks. First, we explore four time series analysis techniques, namely Naive, a baseline technique for time series, Autoregressive Integrated Moving Average (ARIMA), from the statistical field, and Long Short-term Memory (LSTM) and Temporal Convolutional Network (TCN), from the DL family. These techniques were used to forecast the power generation 15 minutes and 24 hours ahead, having as input only power generation historical data. At this point, it was analyzed how training sample size with monthly updates influenced the performance of the forecasting models and which one of them provided less predictive error. Secondly, it was investigated how univariate time series data could be modified by an adversarial attack to decrease models' performance through cross-technique transferability. To this regard, fast gradient sign method (FGSM) attack algorithm, along with a logistic regression substitute model, were used to perform attacks against DL models at test time. Last, we proposed a approach for detecting adversarial attacks during the operation of the plant and defending against them. The detection was based on statistical features and Local Outlier Factor and One-Class Support Vector Machine classifiers. The work was based on real-world data from a solar parking lot plant and these results show that DL models can aid the planning of this plant whereas defense mechanisms against adversarial attacks should be adopted in the context of PV generation forecasting.

Keywords: Time Series Forecast. Deep Learning. Adversarial Machine Learning. Smart Grid. Photovoltaic Energy. Power Generation.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

1-NNDTW 1-Nearest Neighbor Dynamic Time Warping.

ANA Adaptive Normalized Attack.

AR Autoregressive.

ARIMA Autoregressive Integrated Moving Average.

BIM Basic Iterative Method.

CD Critical Difference.

CNN Convolutional Neural Network.

DFA Detrended Fluctuation Analysis.

DL Deep Learning.

FCN Fully Convolutional Network.

FGSM Fast Gradient Sign Method.

I Integrated.

inter Intermittent.

JSMA Jacobian-based Saliency Map Attack.

LOF Local Outlier Factor.

LR Logistic Regression.

LSTM Long Short-term Memory Network.

M Mega = 10^6.

MA Moving Average.

MAE Mean Absolute Error.

Max Maximum.

Min Minimum.

ML Machine Learning.

NN Neural Network.

OCC One-Class Classifier.

OCSVM One-Class Support Vector Machine.

PV Photovoltaic.

ReLU Rectified Linear Unit.

RMSE Root Mean Squared Error.

RNN Recurrent Neural Networks.

sin Sinusoidal.

STD Standard Deviation.

TCN Temporal Convolutional Neural Network.

W Watts (measuring unit of power).

LIST OF SYMBOLS

α	Significance level.
b	Number of stacked convolutional blocks.
b_f	Bias of forget gate.
b_i	Bias of input gate.
b_o	Bias of output gate.
b_c	Bias of cell state.
C_t	Cell state.
d	Dilation convolution skipping factor.
e	Number of differences.
ϵ	Coefficient of adversarial perturbation magnitude.
η	Adversarial perturbation.
f	Generic forecasting model.
F	Legitimate forecasting model.
F'	Substitute forecasting model.
f_t	Forget gate.
h	Forecast horizon.
i_t	Input gate.
i	Input window size.
J	Loss function.
k	Kernel size.
l	Number of stacked layers.
μ	Contamination factor.
n	Number of samples of the test set.
$\nabla_{\mathbf{x}}$	Gradient with respect to $\mathbf{x}$.

ν	Regularization factor in OCSVM.
o_t	Output gate.
p	Number of autorregressive terms.
q	Number of moving averages.
s	Last instance detected as legitimate.
sig	Sigmoidal.
θ	Weights of the adversarial model.
t	Time.
T	Time series length.
$tanh$	Hyperbolic tangent.
W_f	Weight matrix of forget gate.
W_i	Weight matrix of input gate.
W_o	Weight matrix of output gate.
W_c	Weight matrix of cell state.
x_t	Value at time t.
$\hat{x}_{t+h}$	Predicted value at time $t + h$.
ξ_i	Forecast error.
z	Generic variable.

CONTENTS

1 INTRODUCTION

Intelligent generation and distribution of electrical energy are beneficial for systems operators, plant managers and consumers (ANTONANZAS et al., 2016). A key aspect in this process is the accurate forecast of produced energy, which is fundamental to enable the integration of several plants to the grid, save costs, make power grids more reliable amid demand variation, avoid power outage, and prevent plant managers from penalties. It is also advantageous for the sake of the environment (DIAMANTOULAKIS; KAPINAS; KARAGIANNIDIS, 2015), particularly when renewable sources are employed.

For Photovoltaic (PV) generation, the renewable energy source analyzed in this work, forecasting is challenging due to the dependence on meteorological factors such as clouds covering solar panels and variations of solar radiation (TORRES et al., 2018). Thus, building accurate forecasting models based on historical power data is a complex task. Despite this uncertainty in PV generation, a rise in PV systems construction has been observed, which resulted in a 551 MW increase of the Brazilian PV installed capacity only in 2019 (ANEEL, 2019). Improving PV power generation predictability may boost this growth potential, once grid operators would feel more confident about PV generation reliability.

Recent works attempted to predict future PV power output (TORRES et al., 2018; DAS et al., 2018; AKHTER et al., 2019; WANG; QI; LIU, 2019; MELLIT et al., 2020) making use of statistical models, artificial intelligence, or a combination of both. These models are often built in batch, which assumes that data distribution does not change over time. However, updating machine learning (ML) models when dealing with univariate time series (observing a single variable throughout time) forecast according to new incremental observed samples tends to provide better results as the training sample size increases (CERQUEIRA; TORGO; SOARES, 2019).

As PV plants are part of power grids, especially smart grids, having a good forecasting model does not suffice. The plant manager must also worry about the plant susceptibility to different attacks, which seek to interrupt the grid's safe operation or obtain financial gains, for instance (MEHRDAD et al., 2018). Islam, Baig e Zeadally (2019), for example, mention an example of an attack against the communication systems of Ukrainian distribution companies that caused widespread negative consequences for consumers. These consequences ranged from destruction of files to the incapacitation of power supplies. This fact reiterates that power systems can be the goal of adversaries and security aspects related to the grid must be taken into consideration.

Many smart grids functioning rely on the result of ML forecasts and the forecasts

of renewable energy generation are gaining importance in the planning of electrical power systems (DUCHESNE; KARANGELOS; WEHENKEL, 2020). Since wrong forecasts on power generation may drive operators or automated control to make harmful decisions towards grid balancing, it must also be a vulnerability point of attention. For instance, in (MAHMUD et al., 2019), the authors show the impact of prediction error in the power demand (a complementary task to power generation) in a context where a battery energy storage system, a PV system and electric vehicles are connected. The error led to increased battery charging-discharging cycles, which reduced battery life, increased energy cost and caused incorrect energy sharing. This ratifies that besides obtaining accurate models, their reliability should be investigated.

Adversarial attacks are among the threats that may influence the forecast result when using ML-based models. These attacks generate subtle perturbations to the original input of the model aiming at maximizing its prediction error (SZEGEDY et al., 2013). DL models are specially susceptive to this kind of attack as shown in the image and text domains (AKHTAR; MIAN, 2018; HAO-CHEN et al., 2020). Although the usage of DL to deal with real-world time series has increased, including those related to smart grids (ZHANG; HAN; DENG, 2018), few works analyzed the vulnerability of DL models to adversarial time series examples (ALFELD; ZHU; BARFORD, 2016; FAWAZ et al., 2019; ABDU-AGUYE et al., 2020) or adversarial attacks in power systems (CHEN; TAN; DEKA, 2018; NIAZAZARI; LIVANI, 2020).

In this context, we first investigate the performance of traditional time series forecasting techniques with Deep Learning (DL) methods to forecast power generation 15 minutes and 24 hours ahead taking advantage of the training sample size increase. Since the amount of data about power generation increases over time, these models are updated and assessed monthly with the goal to evaluate the influence of the training sample size in their performance. It is made also to obtain the best forecasting model for this plant using only its generation historical data.

As noticed, the prediction error may cause a great impact in the grid operation and adversarial attacks against smart grids models (even more in regression tasks) have been overlooked by the literature. Considering that anticipating the potential model vulnerabilities and analyzing the impact of the possible attacks are the first steps of a proactive protection approach (BIGGIO et al., 2013), we perform adversarial attacks during the test time of the forecasting models to evaluate their robustness under this threat.

Relatively to the attacker, we impose constraints attempting to avoid inaccurate and unrealistic security threats (SUCIU et al., 2018). An adversary with restricted knowledge about the training data and the victim's forecasting models modifies the input data by using Fast Gradient Sign Method (FGSM), a low-cost attack, and a Logistic Regression (LR) substitute model, which is less complex than DL models, to degrade the forecasting

performance.

Given the impact of adversarial attacks in smart grids, we also propose a defense approach to be used during the operation phase of the plant. In the defense, each new instance is first assessed by an external detector and samples classified as adversarial do not follow to the forecasting model.

1.1 Problem and Hypothesis

As observed, there are several issues concerning production forecast and adversarial attacks to PV generation context. In this work, we posed the following research questions:

1) Is it possible to obtain an accurate model for forecasting power generation in the novel PV plant with time horizons of 15 minutes and 24 hours?

2) Can adversarial attacks with limited knowledge and computing power deteriorate the performance of the forecasting model?

3) Are these attacks detectable and how to efficiently tackle them?

These questions lead to the three main hypothesis investigated in this work:

1) Traditional statistical time series methods and DL models can predict PV generation satisfactorily, and increasing the training sample size is beneficial to decrease the model error.

2) FGSM, in a gray-box attack configuration, can degrade the performance of the forecasting models using logistic regression as substitute model and only two months of recorded data.

3) Based on appropriate features, one-class classifiers are able to distinguish between legitimate and adversarial inputs, and replacing the detected adversarial samples by the last detected legitimate sample can reduce the effect of the attacks.

1.2 Objectives

The main objective of this work is developing accurate models for PV power generation forecast and securing it against possible attacks that these models might be subject to. For this, specific goals were defined:

1. Evaluating statistical and DL models performance according to their different hyperparameter configurations;
2. Analyzing the training sample size influence in the models' behavior;

3. Comparing the performance of these models when predicting different time horizons;
4. Investigating the effect of adversarial attacks and different attack patterns on these models;
5. Proposing an approach for identifying the attacks and securing against them.

1.3 Contributions

This mater thesis and the works derived from it bring several contributions, which can be summarized as follows:

1. For years, most of the forecasting studies have aimed only at finding the model with high accuracy, not investigating security aspects of these models (CHEN; TAN; ZHANG, 2019). In this study, both robust forecasts and possible vulnerabilities are addressed;
2. It explores adversarial attacks scenarios in univariate time series, which represents a very underexplored area. Further, it deals with a regression task, which is even a greater gap in literature;
3. The majority of works related to smart grids focus on forecasting the load on the demand side, many times using simulated data. Instead, this work focuses on forecasts of power generation, using a real-world dataset;
4. In fact, as far as we are concerned, this is the first time a work investigates adversarial attacks and defenses to a regression DL model built upon a real-world generation time series;
5. The vulnerability verification is followed by a defense approach proposal to detect adversarial attacks;
6. It supposes limited knowledge about the victim and an attacker with limited computational resources, getting closer to the limitations of realistic adversaries and shedding light on cross-technique transferability using a simple model as a substitute model to more complex ones.

1.4 Outline

This master's thesis is organized as follows. In Chapter 2, it is provided the background related to time series, the forecasting methods adopted in this work and adversarial machine learning. Chapter 3 presents how this work relates to previous works in the literature. Chapter 4 exposes the proposal for detecting adversarial attacks in power

generation forecast. Chapter 5 describes the dataset and experimental details. Chapter 6 reports and discusses the results of the experiments, communicating the forecasting performance of the models, the impact of the attacks and the proposed defense. Chapter 7 shows the conclusions and future research that may be developed continuing this master's thesis.

Additionally, two appendices are provided with complementary experimental results (Appendix A) and calculation (Appendix B).

2 THEORETICAL BACKGROUND

In this chapter, the main theoretical concepts explored in this master's thesis are presented. It begins with the definition of time series. Afterward, the statistical and DL methods regarding time series forecasts are provided. To conclude the chapter, fundamental concepts about adversarial machine learning are dispensed.

2.1 Time Series

A time series is given by a data sequence in a particular time period, and this data can produce different values at distinct moments in time. Formally, it can be defined as a chronologically ordered set of observations $X = [x_1, x_2, \dots x_T]$ in which T corresponds to the length of the series (PARZEN et al., 1961).

The forecasting task consists of finding a function f that predicts the h-th future value in any time t, i.e., $\hat{x}_t + h$ based on i past values:

$$\hat{x}_{t+h} = f(x_{t-(i-1)}, x_{t-(i-2)}, \dots, x_{t-1}, x_t), \quad i \leq t, \tag{2.1}$$

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 should also attend the condition $t \leq T - h$. Figure 1 illustrates a time series with $T = 10$ and a function f with $i = 4$ and $h = 1$. For this, $4 \leq t \leq 10 - 1$. Note that when $t = 4$, the input window is composed by x_1, x_2, x_3 and x_4 and the forecast is $\hat{x}_5$.

In addition, time series can present seasonality. This 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.

The Naive method (HICKMAN, 1942), also known as persistence method, is the most straightforward method to forecast the future value in a time series. For each t, it considers the current observed value x_t as the forecast of the h-th next value of a time series:

$$\hat{x}_{t+h} = x_t \tag{2.2}$$

Its greatest advantage in relation to other methods is the simplicity of computa-

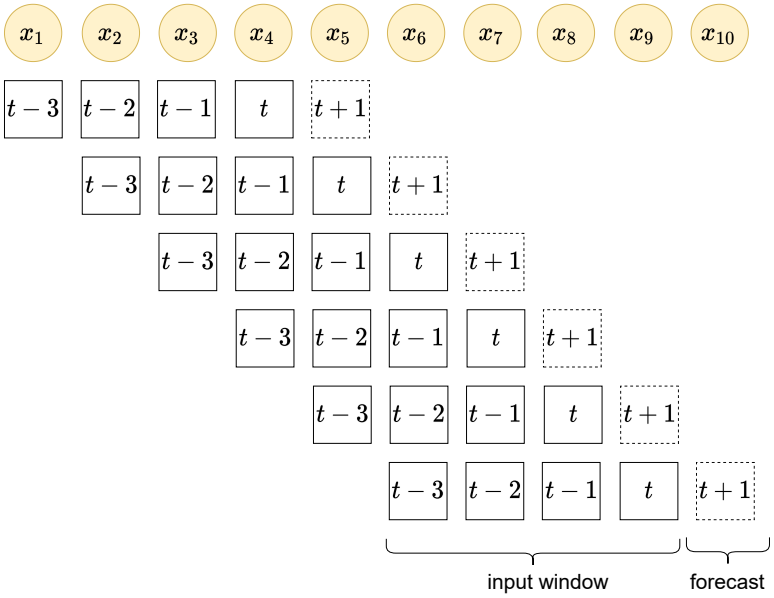


Figure 1 – Example of time series with $T = 10$, $i = 4$ and $h = 1$.
(Source: the author.)

tion. However, this method assumes there are no changes between $t+h$ and t and its result correspond to the worst permissible error, in such a way that a model with worse performance than Naive should be disregarded (MCLAUGHLIN, 1983). Given the complexity involving PV power generation (for instance seasonality and meteorological factors) and the need for anticipating changes, more sophisticated models are usually required.

2.1.1 ARIMA Method

Understanding the different factors of a time series is important to extract information that can be used to predict future points in this series. Many statistical methods were applied in time series for this purpose and one of the most adopted, ARIMA, is presented next.

The ARIMA method is essentially exploratory and seeks to fit a model to adapt to the data structure (BOX; JENKINS, 1990). The term ARIMA is a combination of three components:

- Autoregressive (AR) indicates that the evolution variable of interest is returned to its previous values.
- Moving Average (MA) indicates that the forecast error consists of a linear combination of past respective errors.
- Integrated (I) indicates the process of differentiating between current values and previous values.

With the aid of the autocorrelation and partial autocorrelation functions, it is possible to obtain the essence of the time series so that it can be modeled. Then, information such as trends, variations, cyclical components, and even patterns present in the time series can also be obtained (HO; XIE, 1998). This allows the description of its current pattern and predictions of future series values (PENA et al., 2014).

To increase the performance of ARIMA when dealing with non-stationary data, the integrated part (I) is applied, whereas differentiation processes are carried out and can be applied more than once until stationarity is obtained. This model is defined by the values (p, e, q), where p is the number of auto-regressive terms, e is the number of differences, and q is the number of moving averages.

The general form of ARIMA can be expressed as (KOTU; DESHPANDE, 2019):

$$x'_t = I + \beta_1 x'_{t-1} + \beta_2 x'_{t-2} + \cdots + \beta_p x'_{t-p} + \xi_t + \phi_1 \xi_{t-1} + \phi_2 \xi_{t-1} + \cdots + \phi_q \xi_{t-q} \qquad (2.3)$$

where ξ_i is the forecast error at point i, ϕ and β are the coefficients that need to be learned from the training data and $x'_t = x_t - x_{t-1}$ represents the difference between consecutive data points.

The greatest disadvantage of ARIMA is assuming that future values are linearly related to current and past values of the time series, as well as to white noise (KHASHEI; BIJARI, 2011). Often, real-world problems have a nonlinear nature (ENNS, 2010) and then this method might not be appropriate.

2.2 Deep Learning

ML, particularly DL, models have been showing to be adequate for dealing with time series made up of power-related data, from both demand and generation sides (MELLIT et al., 2020; SHAO et al., 2020; BOMFIM, 2020). Alongside the great performance achieved by the DL methods, one of their most notable advantages is the capability of processing data in their raw form (LECUN; BENGIO; HINTON, 2015). Among the most explored DL models for time series in power systems, TCN and LSTM achieved relevant results for this type of data (TORRES et al., 2018; YEN et al., 2019; LARA-BENÍTEZ et al., 2020).

2.2.1 Temporal Convolutional Network

TCN (LEA et al., 2017) is a specific kind of Convolutional Neural Network (CNN) that has the capability of dealing with time series. TCN can be understood as a 1-

Dimension CNN architecture that uses causal convolution (BAI; KOLTER; KOLTUN, 2018).

Let k be the number of input values for the convolution operation, i. e., the kernel size. In standard convolution, the input values for the convolution operation corresponds to the k most proximal values, which would include future values of the time series. On the other hand, the convolution operation does not depend on future values for outputting a value at time t in causal convolution. In such a way, causal convolution prevents leakage from future information to the past.

The causal convolution may be submitted to dilation. In TCN, dilated convolutions in one dimension are used seeking to explore long-term patterns. According to a dilation factor d, the procedure for doing this is skipping $d-1$ values between the inputs of convolution. For example, if $d = 1$, no input values will be skipped and corresponds to the convolution without dilation; if $d = 2$, one observed input value is followed by a skipped value. Since each layer might have different d values, the dilations will be denoted in this work as $[d_1, d_2, \ldots, d_n]$, where d_1 corresponds to the dilation rate of the layer that is the closest to the input, and d_n to the layer that is the closest to the output.

Figure 2 compares standard convolution, causal convolution without dilation and dilated convolution, all with k=3. Note that in standard convolution (Figure 2a) used in conventional CNN, unseen values at time t (x_{t+1} and x_{t+2}) would be required for computing y_{t+1}. Instead, causal convolution (Figures 2b and 2c) look up only at the current and past values.

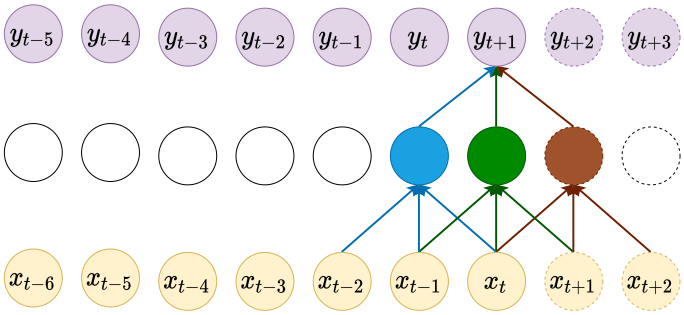
Comparing the blocks that perform causal convolution, Figure 2b) shows that without dilation, the 5 most recent observations (x_{t-4} to x_t) influence y_{t+1}. When using dilation, Figure 2c), the 7 most recent observations (x_{t-6} to x_t) will be considered for outputting y_{t+1}, allowing the exploration of longer-term patterns than causal convolution without dilation.

Still aiming to increase the ability of the network to explore long-term patterns, b convolutional blocks can be stacked. Since it increases the number of hyperparameters, the learning process becomes more complex. Figure 3 shows a dilated causal network with two stacked blocks. In this case, the 13 most recent observations (x_{t-12} to x_t) will be considered for outputting y_{t+1}.

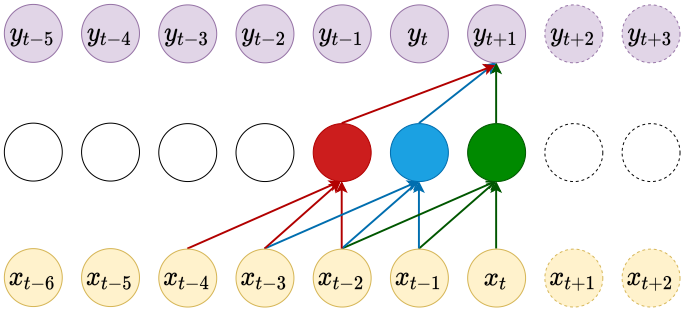
The variables b, k and d define the receptive field of the network. It can be expressed according to Equation 2.4.

$$receptive\ field = b \times k \times d_n \tag{2.4}$$

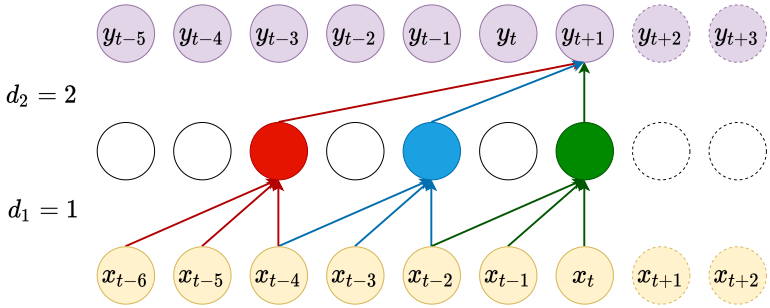
This equation acts as a constraint when defining possible hyperparameters for the



a) Standard convolution



b) Causal convolution without dilation



c) Dilated causal convolution with d=[1,2]

Figure 2 – Comparison of convolution blocks composed by two convolutional layers and k=3.

(Source: the author.)

network. For an adequate use of TCN, the receptive field should cover seasonality in past history.

2.2.2 Long-Short Term Memory Network

Recurrent Neural Networks (RNN) is a classical class of neural network for dealing with time series. In this network, the information is propagated throughout a chain of

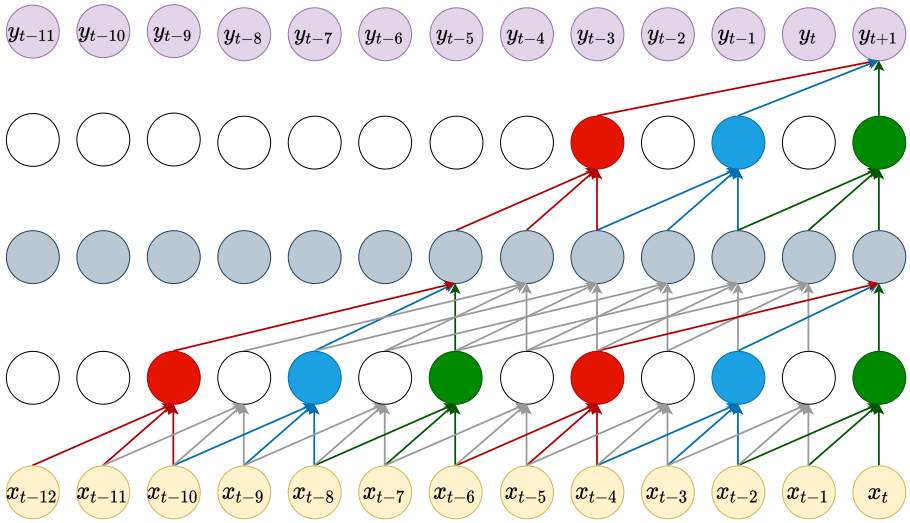


Figure 3 – Example of dilated causal networks with b=2, k=3 and d=[1,2].
(Source: the author.)

repeating units (**U**) of a neural network located in the hidden layer. Figure 4 exemplifies this structure.

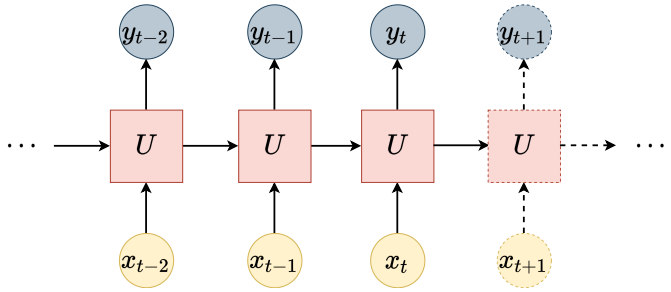


Figure 4 – General structure of a RNN.
(Source: the author.)

In standard RNN, each unit **U** is composed of a simple neural network, for instance, a single layer perceptron. However, standard RNN suffers from error backflow problems such as gradient vanishing and explosion, which restrain long-term dependency learning. The first may lead to very slow learning, and the latter, to weight oscillation.

The backflow calculation drawback was one of the main motivations for the development of LSTM (HOCHREITER; SCHMIDHUBER, 1997). This network, a particular case of RNN, uses a gating approach for learning long-term dependencies without losing the short-term capability.

These gating mechanisms are inside memory cells, which are located in the hidden layer. Each unit in traditional LSTM has the aspect presented in Figure 5.

The hyperbolic tangent function ($tanh(.)$, Equation 2.5) transforms the values to the range from -1 to 1 and the sigmoid function ($sig(.)$, Equation 2.6) to values between 0

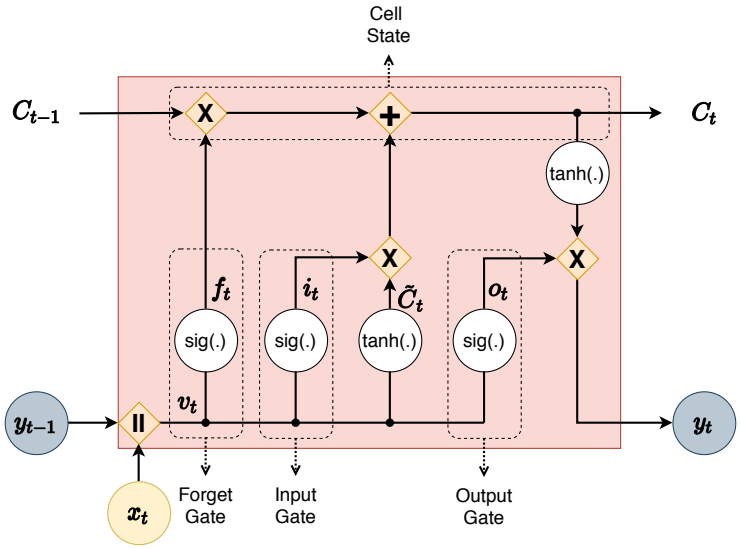


Figure 5 – Elements of a LSTM unit.
(Source: the author.)

and 1. This property makes $sig(.)$ to act as a gate, since the values that are transformed to 0 are forgotten by the network and the values that are transformed to 1 are kept.

$$tanh(z) = \frac{2}{1 + e^{-z}} - 1 \tag{2.5}$$

$$sig(z) = \frac{1}{1 + e^{-z}} \tag{2.6}$$

Each cell is mainly composed of three gates (forget, input and output gates) and a cell state. Let v_t be the concatenation of x_t and the output of the the previous unit y_{t-1}. W_f, W_i, W_o and W_c correspond to the weights matrices related to forward, input, output gates and cell state; b_f, b_i, b_o and b_c correspond to bias of the respective gates and cell state.

The forget gate (f_t) is the main gate to select which information should pass forward to the next cell or be eliminated (Equation 2.7).

$$f_t = sig(W_f.v_t + b_f) \tag{2.7}$$

The input gate (i_t) is used to update the cell state and to select what will be written to the cell (Equation 2.8).

$$i_t = sig(W_i.v_t + b_i) \tag{2.8}$$

The output gate (o_t) decides the values that will be part of the output (Equation 2.9).

$$o_t = sig(W_o.v_t + b_o) \tag{2.9}$$

The cell state (C_t) allows the gradient flow and is updated by Equation 2.10.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.10}$$

$\tilde{C}_t$ selects the new candidate values that can be added to regulate the network (Equation 2.11).

$$\tilde{C}_t = tanh(W_c.v_t + b_c) \tag{2.11}$$

This value is combined to i_t by a pointwise multiplication operation forming $i_t * \tilde{C}_t$. Then, the output of the cell at time t is calculated by Equation 2.12.

$$y_t = o_t * tanh(C_t) \tag{2.12}$$

2.3 Adversarial Machine Learning

Solutions that depend on ML to work might suffer attacks from adversarial examples (SZEGEDY et al., 2013). These are intentionally perturbed examples that aim to maximize the prediction error of models. Accepted hypothesis for the vulnerability of ML models towards adversarial examples are the limitation of training data to cover all the problem and limited generalization of the models (ROUANI et al., 2019).

One alternative for improving the security of these ML-based solutions is anticipating the attacker by adopting a proactive security design (BIGGIO; ROLI, 2018). This design consists of the following main steps:

- Investigating the attacks that the system might face;
- Simulating these attacks;
- Evaluating the impact that these attacks might cause to the system;
- Implementing effective countermeasures to the relevant attacks.

For achieving the first step, creating possible adversarial examples is desired. This process can be modeled according to the attacker's goal, capability and knowledge.

2.3.1 Adversary's Goal

This part of the threat model indicates the purpose of the attacker when performing an adversarial attack. It can be classified according to the security violation, attack specificity and error specificity (BIGGIO; ROLI, 2018).

First, the attacker may violate the security of the system in availability, integrity, privacy and confidentiality (PAPERNOT et al., 2018):

- Availability: the attacker compromises the functionalities of the system. In this type of security violation, the system reduces its quality and performance, or obstructs the access of legitimate users to the system (e.g., a denial of service).
- Integrity: it produces system outputs or behavior desired by the attacker without necessarily compromising the functionalities (e.g., skipping the alarms of intrusion).
- Confidentiality and privacy: attacks are performed to obtain private information about the system or its related data. It includes accessing details about used models and their configuration, and also to possibly sensitive knowledge.

The attack may also be classified according to the specificity. In this sense, it can be targeted, when only specific instances of the system (e.g., specific users, specific periods of time) are aimed; or indiscriminate, when any instance is aimed by the attacker.

Singularly for classification tasks, it is also possible to categorize the attacker according to error specificity. It can be specific if it aims a misclassification to a specific class; or generic if it aims a misclassification to any class but the true class.

2.3.2 Adversary's Capability

If the attacker can access and modify training data, the influence of the attacker is said to be causative. Attacks that can modify the training data are known as poisoning attacks (JAGIELSKI et al., 2018).

On the other side, if the attacker can modify only test data, the influence is said to be exploratory. For classification problems, attacks that occur at test phase are known as evasion attacks (TABASSI et al., 2019).

2.3.3 Adversary's Knowledge

The attacker's knowledge might differ according to the access to the system components set: the training data, features space and the learning algorithm. The latter may also involve the knowledge of the loss function and the associated parameters/hyperparameters (BIGGIO; ROLI, 2018).

A perfect-knowledge attack, also known as a white-box attack, implies that the attacker has access to the entire set of components. It represents the worst scenario for the target since the attackers might use all the mentioned components in their favor, representing an upper-bound of performance deterioration.

A zero-knowledge attack, also known as a black-box attack, implies that the attacker does not have substantial knowledge about the system components. However, the attacker might be able to input information and get feedback on the output corresponding to a known input.

A limited-knowledge attack, also known as gray-box attacks, lies between the previous attacks. For this purpose, it may have partial access to the training data, knowing the training algorithm or the feature space.

Specifically for the case in which the attacker does not have knowledge about the learning algorithm, it has to define 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 (SZEGEDY et al., 2013).

In this context, the attacker train a substitute model F' and creates adversarial examples for F'. Later, the attacker feeds the victim's model F with the malicious data initially created for F'. There are two main reasons for transferability to work: error spaces for each model are high-dimensional, making intersection between them probable; and two different models designed for the same problem that have similar performance might have similar decision boundaries (PAPERNOT et al., 2018).

2.3.4 Adversarial Examples Generation

Based on their knowledge, the attackers must generate effective samples to fool the model. Ideally, the perturbation caused by the attack must be close to the original example and imperceptible to a human (YUAN et al., 2019).

These attacks can be executed at one step or iteratively. The latter tend to perform better than the former one, however, it requires more computational time and interactions with the victim's model.

Most of the methods for crafting adversarial examples were originally designed for images. Fast Gradient Sign Method (FGSM) (GOODFELLOW; SHLENS; SZEGEDY, 2014) is one of the most notable methods, and is the basis for many later methods (ROZSA; RUDD; BOULT, 2016; KURAKIN; GOODFELLOW; BENGIO, 2016; DONG et al., 2018; TRAMÈR et al., 2017). In the original proposal, the gradient is updated in the direction of each pixel sign.

The perturbation η generated by FGSM is given by Equation 2.13:

$$\eta = \epsilon * sign(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y)) \tag{2.13}$$

where ϵ corresponds to the coefficient that controls the perturbation magnitude, $\nabla_{\mathbf{x}}$ to the gradient of the associated function with respect to $\mathbf{x}$, $\mathbf{x}$ to the input to the model, y to the output associated to $\mathbf{x}$, θ to the weights of the adversarial model and $J(.)$ to the loss function. The $sign(.)$ function assumes the values of +1, 0 or -1 according to its argument (in this Eq. 2.13, the direction of the gradient):

$$sign(z) = \begin{cases} -1 & \text{if } z < 0; \\ 0 & \text{if } z = 0; \\ +1 & \text{if } z > 0. \end{cases}$$

The FGSM attack algorithm is computationally cheap and able to mislead the model. Figure 6 is an example of how this algorithm is suitable to generate adversarial samples. The figure on the left is originally classified as a panda with 57.7% of confidence. By adding a very small perturbation, with $\epsilon = 0.07$, the image on the right is classified as a gibbon with 99.3% of confidence.

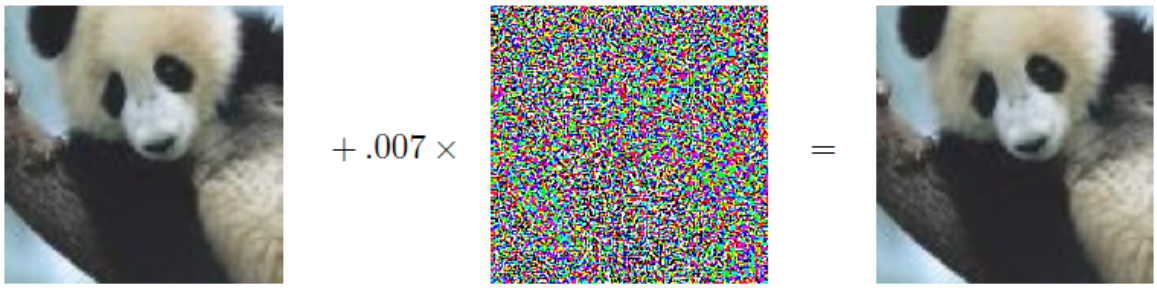


Figure 6 – Example of how FGSM perturbation can mislead the model from a panda (left) to a gibbon (right).
(Source: (GOODFELLOW; SHLENS; SZEGEDY, 2014))

2.3.5 Adversarial Defenses

Countermeasures against adversarial examples fall into three main strategies in the image literature: gradient masking/obfuscation, robust optimization and adversarial examples detection (HAO-CHEN et al., 2020):

- Gradient masking/obfuscation intends to hide the gradient of the model, given the relevance of this information for many attack algorithms. Gradient masking defenses examples include network distillation (HINTON; VINYALS; DEAN, 2015; PAPERNOT et al., 2016; PAPERNOT; MCDANIEL, 2017), in which the DL architecture

is reduced after training, and randomization (XIE et al., 2017; DHILLON et al., 2018; LIU et al., 2018), in which parts of the model or of the input are randomly removed during inference time;

- Robust optimization influences the learning of the models. Adversarial retraining (GOODFELLOW; SHLENS; SZEGEDY, 2014; MADRY et al., 2017; TIAN et al., 2019) is the outstanding approach of this category. It consists of including adversarial examples during the training of the model. By including both legitimate and attacked samples, the models tends to be more robust against the attacks. However, it requires training the model with adversarial examples of every different attack (WANG et al., 2019), which leads to a practical limitation;

- Adversarial examples detection check the consistency of the results, for instance with denoising (XU; EVANS; QI, 2017; MENG; CHEN, 2017), in which the input is modified in features that normally will not change the prediction result. It is also possible to design an external model to detect samples as benign or malicious examples and prevent the model to have access to adversarial samples during test time (LU; ISSARANON; FORSYTH, 2017; METZEN et al., 2017; FEINMAN et al., 2017; GROSSE et al., 2017).

The latter strategy will be used as defense in this work. The motivations for this choice will be better explored in Chapter 4.

2.4 Chapter Conclusion

In this chapter, the main concepts related to this work were described. First, time series was presented. These data will be the input of the forecasting models, so after that, the Naive and ARIMA methods, as well as LSTM and TCN, were further described. Afterwards, adversarial ML was introduced. In the next chapter, related work is presented.

3 RELATED WORK

We present, in this chapter, related work in the literature that addresses time series forecasting and electricity generation forecasting. The chapter also exposes related work regarding adversarial attacks in smart grids and time series, and contrasts ours to previous works.

3.1 Time Series Forecast in Electric Power Systems

The great majority of works in the literature about electrical related forecast is focused on the consumption (load) side. Nevertheless, several works attempted to accurately predict renewable energy production, with forecast horizons varying from the next minutes to the next days. These methods follow four general groups: physical, statistical, artificial intelligence and hybrid methods (HODGE et al., 2018; MELLIT et al., 2020):

- Physical methods are based on numerical methods or satellite images (DOLARA; LEVA; MANZOLINI, 2015). These methods generally are more adequate for long time horizons (usually greater than one month) and do not need historical operation data. However, their implementation is hard, since it relies on the plant construction parameters and physical information that are given by hard-to-access equipment.
- Statistical and probabilistic methods such as regression analysis (WANG; SU; SHU, 2016), adaptive filters (YANG, 2019) and time series (NOBRE et al., 2016) consist of mapping the relation between input and output of the data. These methods tend to be more adequate for short-time horizon (up to 1 day) and their implementation usually needs including descriptors. These methods do not require the full knowledge about the plant operation, delivering simpler and more universal methods than physical methods, but they require high computation costs and correct annotated data (WANG; QI; LIU, 2019).
- Artificial intelligence, particularly ML, methods show the advantage of not requiring information about the PV system construction. They rely, instead, on previous registered data. Another advantage of these methods is the capacity to deal with complex data characteristics, as non-linearity (DAUT et al., 2017).
- Hybrid methods generally combine the previous categories and tend to present good results due to the combination of advantages (DOLARA et al., 2015).

Regarding statistical methods, Atique et al. (2019) used ARIMA to forecast the total daily solar energy generated in a specific solar panel. In most of recent works, this

method is compared to DL. For instance, Jaihuni et al. (2020) compared ARIMA and LSTM to a hybrid version of ARIMA and LSTM to predict 5 minutes and one hour ahead generation. The hybrid version had better performance in predicting the longest forecast horizon, whereas LSTM and ARIMA outperformed for 5 minutes ahead.

Another example of solar energy generation prediction through DL methods is seen in Torres et al. (2018). In this work, the application of DL consists of predicting the generation of energy for the next day. According to the authors, the proposed method was capable of handling big time series data. In fact, as shown in Wang et al. (2019), DL offers very competitive advantages when predicting renewable energy: these methods offer predictive performance, do not need manual feature extraction and are able to deal with huge data volumes.

In Wang, Qi e Liu (2019), the authors evaluated CNN, LSTM, and a hybrid model with CNN and LSTM for modeling a PV system. The results showed robustness, stability, and great performance. TCN is also a very suitable DL method, as shown in (YEN et al., 2019), in which Yen at al. verified that TCN is capable of satisfactorily predicting PV generation and for fault diagnosis.

3.2 Adversarial Machine Learning in Smart Grids

With respect to adversarial attacks in smart grids and time series, Chen, Tan e Deka (2018) evaluated adversarial attacks to feed-forward NN and RNN models for simulated data on power quality classification and building load forecasting, respectively. Based on the results, the authors stimulated more discussions for increasing the robustness of the models implemented in power systems.

Fawaz et al. (2019) adapted FGSM and Basic Iterative Method (BIM) to univariate time series classification and performed attacks to DL models. These attacks achieved an average reduction in the model's accuracy of 43.2% and 56.89%, respectively, and showed that FGSM allows real-time adversarial sample generation. The authors of this work claim that it is the first work to consider the vulnerability of DL models in relation to adversarial time series examples.

Tian et al. (2019) explored a proposed attack named Adaptive Normalized Attack (ANA) and FGSM to generate adversarial signals to a Neural Network (NN) classifier of power quality. They also investigated the usage of adversarial training as defense approach.

Niazazari e Livani (2020) performed attacks to a multiclass CNN trained in simulated data. The target model classifies events as line energization, capacitor bank energization and faults in the grid. The attacks were generated using FGSM and Jacobian-based Saliency Map Attack (JSMA) algorithms.

Karim, Majumdar e Darabi (2020) proposed using adversarial transformation network to attack 1-Nearest Neighbor Dynamic Time Warping (1-NNDTW) 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 robustness of the models.

Abdu-Aguye et al. (2020) proposed using One-Class Support Vector Machine (OCSVM) to classify samples as original or perturbed using as basis the attacks and datasets presented in (FAWAZ et al., 2019). The authors claimed to reach 90% detection accuracy on most datasets and up to 97% in the best case.

Table 1 presents a comparison among these works in the main aspects and our proposal. We compared them in terms of dataset description, victim's model, kind of problem, attack algorithms, the usage of transferability and defense approach.

Table 1 – Adversarial ML works in smart grids.

Reference	**Dataset**	**Model**	**Output**	**Attack**	**Transf.**	**Defense**
Chen et al. (2018)	Simulated data on power quality and building load	NN and RNN	Classification and Regression	FGSM	Yes	–
Favaz et al. (2019)	Several univariate time series, including power demand	ResNet	Classification	FGSM and BIM	Yes	–
Tian et al. (2019)	Simulated data on power quality	NN	Classification	FGSM and ANA	–	Adversarial training
Niazazatri et al. (2020)	Simulated data of grid events	CNN	Classification	FGSM and JSMA	–	Adversarial training
Karim et al. (2020)	Several time series datasets including power demand	1-NN DTW and FCN	Classification	FGSM and ATN	Yes	Adversarial training
Abdu et al. (2020)	Several univariate time series, including power demand	ResNet	Classification	FGSM and BIM	–	OC-SVM
Proposed (including (SANTANA et al., 2020))	Real-world PV generation time series	TCN and LSTM	Regression	FGSM	Yes	OC-SVM and LOF

Regarding the datasets, the previous works alternate between using simulated

data or real-word data. However, none of them tackled real-world generation data, as this work. Analyzing the models, NN and Deep NN are the primary victim's model goal. These models are mostly classifiers, except by (CHEN; TAN; DEKA, 2018) and our work.

All these works adopt at least FGSM as attack algorithm, showing the importance of this attack algorithm in adversarial ML literature. This was a motivation for the choice of this attack in this work.

The consideration of limited knowledge for the attack and, consequently, the need for transferability were not addressed in all works, but it is relevant since hardly the attacker will have access to training data, features space and the learning algorithm/(hyper)parameters in a real-world scenario.

When observing the defense strategies, a limited number of works propose adversarial defenses and most of them chose adversarial training. Using a detector was addressed in (ABDU-AGUYE et al., 2020), but the authors considered only OCSVM and in a classification forecasting task. As it will be shown in this work, we consider both OCSVM and Local Outlier Factor (LOF) and a defense for adversarial attacks in the power generation regression scenario.

The publication dates of these works reveal that Adversarial ML in Smart Grids is a novel and growing research area. This master's thesis contributes to this research area, exploring unattended gaps and contributing to the understanding of this kind of attack.

3.3 Chapter Conclusion

This chapter showed how this work is related to previous ones. As observed, DL has been used and tends to produce satisfactory results to forecast smart grid-related data. However, their performance can be damaged by specially designed samples when predicting univariate time series. The main differences between these works and ours were underlined. In the next chapter, our approach for securing the parts of the plant that depend on the forecasting results is described.

4 ADVERSARIAL ATTACKS DETECTION AND DEFENSE PROPOSAL

In the previous chapters, we noted the hazard that adversarial examples can cause. Considering the lack of defenses against adversarial attacks to regression models and the deteriorating influence of adversarial attacks to the forecasts results, in this chapter we propose a detection and defense approach during test time in this kind of problem.

4.1 Approach Components

As shown in Subsection 2.3.5, a defense approach against adversarial attacks applied in other domains is designing an external detector and preventing the attacked model from having access to adversarial samples during test time. Although the trend of adversarial ML in smart grids is using adversarial retraining (as depicted in Table 1), ML was successful to detect attacks in other domains and is promising to also detect attacks in time series and smart grids (ABDU-AGUYE et al., 2020; ZARPELÃO et al., 2020). Consequently, ML classifiers might be successful detectors of adversarial attacks in the domain of PV generation and we elected this kind of detection.

The main motivation is the following: if new possible adversarial attacks are found, adversarial retraining would require a new training of the whole forecasting model to include this type of attack; instead, using a detector would demand the re-training of only the detector model, which is generally simpler than the forecasting model and consequently lighter to train. Further, the detector model might be trained using examples only from the normal class, which reduces the need of a detector update as new attacks are found.

The proposed approach for overcoming adversarial attacks is shown in Figure 7. It consists of three main blocks. Briefly, during the test time the time series is windowed with size i as new data arrive to the time series (block 1). For each window, 11 features are extracted (block 2). These features will serve as input to the ML-based detector in block 3. If the detector does not identify the input as an attack, the corresponding original window will be then inputted to the forecasting model. Otherwise, the most recent instance predicted as legitimate (here denoted as instance s) will follow to the forecasting model.

The statistical features extracted for each input window in block 2 are: Minimum (Min), Mean, Median, Maximum (Max), Standard Deviation (STD), Ratio between Mean and Maximum, Ratio between Minimum and Maximum, Entropy, Correlation, Detrended Fluctuation Analysis (DFA) and Hurst Exponent. These features are light-weight computed, so that do not compromise the performance of the detection approach. The

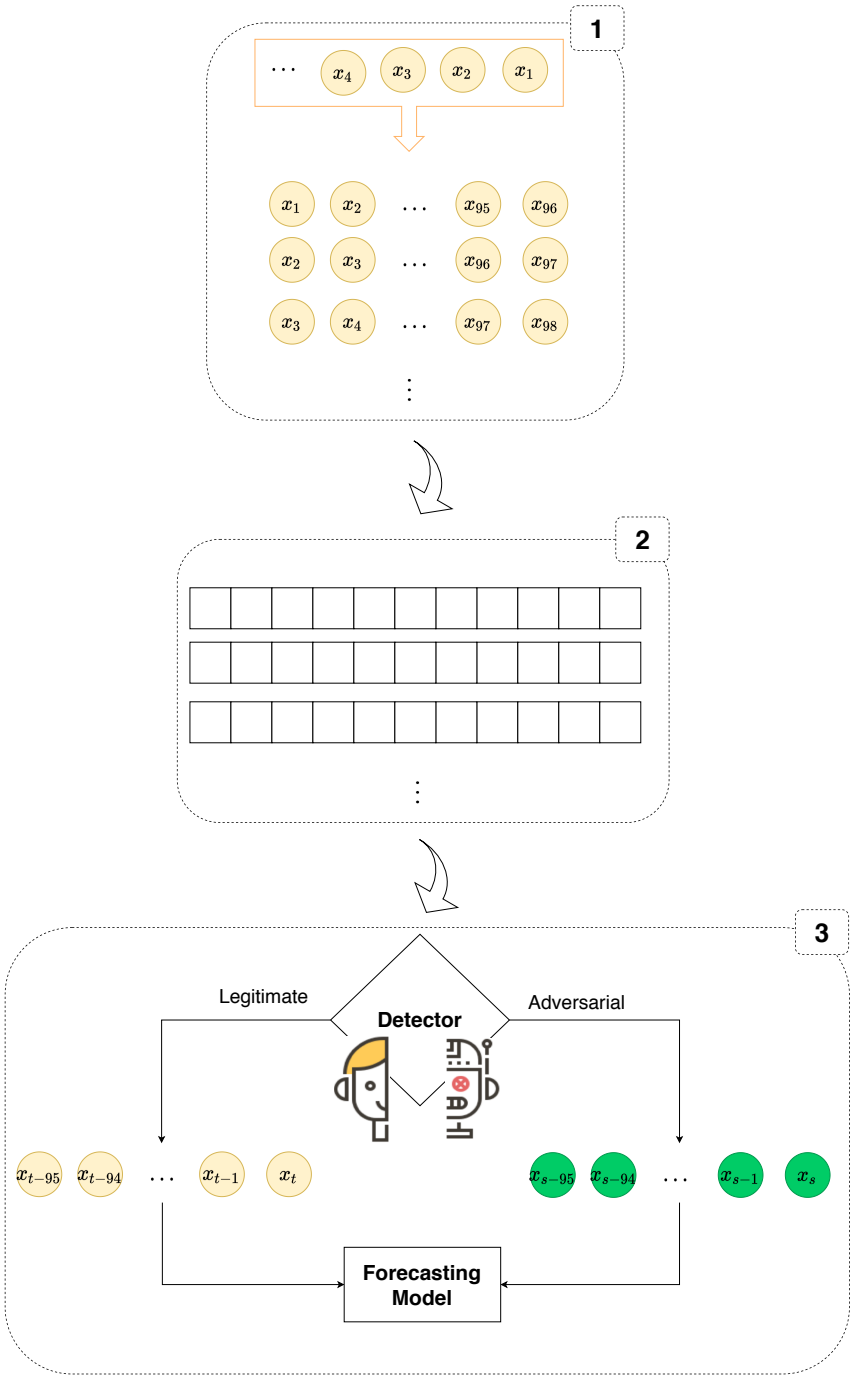


Figure 7 – Approach for adversarial detection and defense at test time.
(Source: the author.)

adoption of more complex features as sample entropy and DFA analysis were motivated by the desire to reach better detection performance, and their use in previous related work to reveal differences between legitimate and adversarial attacks (ABDU-AGUYE et al., 2020).

To build the detector, One-Class Classifiers (OCC) used for anomaly detection were investigated. This decision is based on the hypothesis that adversarial examples can be understood as intentional anomalies (BULUSU et al., 2020). They are simpler models

than TCN and CNN, and are trained based upon normal data (in our case, only with legitimate data). For this reason, they present the additional advantage of not needing to explicitly define every single adversarial sample and might work for attacks beyond the ones investigated in this work (KHAN; MADDEN, 2014). Two OCC, with different biases, will be examined as detectors.

OCSVM (SCHÖLKOPF et al., 2001) is an extension of Support Vector Machine for unlabelled data. For this reason, the non-attacked training data is used to learn the model boundaries. During the training of OCSVM, the ν hyperparameter works as a regularization factor, setting a lower bound of the proportion of training samples used as support vectors and an upper bound on the proportion of training error. In the test time, points that are classified inside the model's boundary are considered as a legitimate input; otherwise they will be considered as adversarial.

LOF (BREUNIG et al., 2000) detects an outlier based on local density. The local density of a sample is compared to the local density of its neighbors. If the analyzed sample's density is considerably lower than the neighbors density, the sample will be considered as adversarial.

When these detectors classify an input as adversarial, which would result in increased error, their classification results will prevent the forecasting model to have access to attacked data in the proposed approach. Meanwhile, as the attacked instance will be replaced by instance s, it promotes an automatic defense and avoids stops in the operation when an attack is detected.

4.2 Chapter Conclusion

This chapter showed the main parts of the proposed approach for detecting and defending the forecasting model against adversarial attacks. In the next chapter, the experimental setup to verify this and other aspects in the studied PV generation problem is described.

5 EXPERIMENTAL SETUP

The experiments of this chapter were split into three setups to emphasize the main segments of this work: assessing the models' performances in terms of error, verifying their vulnerability to adversarial attacks and validating the defense approach against these attacks. Table 2 shows an overview of the main aspects of the experimental setups.

Table 2 – Experimental setups overview.

Aspect	**Setup 1**	**Setup 2**	**Setup 3**
Associated Research Question	Is it possible to obtain an accurate model for forecasting power generation in the novel PV plant with time horizons of 15 minutes and 24 hours?	Can adversarial attacks with limited knowledge and computing power deteriorate the performance of the forecasting model?	Are these attacks detectable and how to efficiently to overcome them?
Forecasting Models	Naive, ARIMA, LSTM, TCN	LSTM, TCN	LSTM, TCN
Evaluation Procedure	Prequential in blocks	Hold-out	Hold-out
Adversary's Knowledge	—	Gray-box (with limited knowledge about training data)	Gray-box (with limited knowledge about training data)
Adversary's Goal	—	Indiscriminate attack	Targeted attack
Adversary's Capability	—	Attacking the data during test time using FGSM	Attacking the data during test time using FGSM
Adversary Defense	—	—	Using OCC as detectors at test time

The code related to the experiments was implemented using Python 3.6. The two first experiments (Setup 1 and Setup 2) were performed using Intel Xeon CPU @2.3GHz and Tesla K80 GPU made available by Google Colab. Setup 3, in turn, used Amazon Web Service platform.

5.1 Setup 1 - Obtaining the Best Forecasting Model

Setup 1 is dedicated to assess the prediction performance of Naive, ARIMA, LSTM, and TCN methods. Naive was picked to be a baseline for our experiment. ARIMA

is likely the most adopted option when it comes to statistical methods for time series analysis. Lastly, LSTM and TCN are DL methods, which represent the state-of-the-art of ML methods for time series forecasting.

It is the first work analyzing the data from the solar parking lot plant installed at the State University of Londrina. It was inaugurated in November 2019 as part of an Energy Efficiency Project approved in the Public Call (Copel-VPDE 001/2017), promoted by the Brazilian National Agency of Electrical Energy. It started to operate in November 2019 with a total capacity of 300 kW distributed over 6 inverters. For being representative of the other inverters, inverter 1 was chosen for the analysis. Since the PV plant relies on solar energy to operate, it is normally turned off between 7 pm and 6 am (of the next day), when the solar irradiance tends to be less usable. Table 3 shows the data description for each month used in Setup 1.

Table 3 – Main monthly information about power generation in inverter 1.

Month	**Maximum value [W]**	**Minimum value [W]**	**Average [W]**	**Number of samples**
2019-11	41521.48	0	8749.14	2880
2019-12	41148.77	0	7582.21	2976
2020-01	42189.33	0	8062.55	2976
2020-02	41194.93	0	7485.98	2784
2020-03	43521.93	0	8506.43	2976

Samples were collected every 15 minutes, implying that each day is composed of 96 samples. Considering this and that the data behavior mostly repeats every day, the input window size was set to 96 ($i = 96$) to cover daily seasonality.

The interest in energy production forecasts can be part of different strategies in very short and short-term time horizons. Thus, the performance of these methods when forecasting the production in the next 15 minutes ($h = 1$) and 24 hours ($h = 96$) is evaluated.

Prequential evaluation in blocks with a growing window (MODHA; MASRY, 1998) was used to show the evolution of performance as the sample size grows and to simulate a situation in which the model is updated monthly. In this case, each month (except the first and the last) was used for test before being incrementally used for training, optimizing the use of available data, as observed in Figure 8. Furthermore, for hyperparameter tuning, 20% of the training data was reserved for validation.

This evaluation strategy is the best prequential approach when compared to prequential approaches using sliding windows, gaps between train and test or growing the training set every new instance (CERQUEIRA; TORGO; MOZETIČ, 2020). Besides, it uses less computation resources than repeated hold-out.

A preprocessing step with z-score was performed to improve convergence. The normalization parameters were obtained for each new training set.

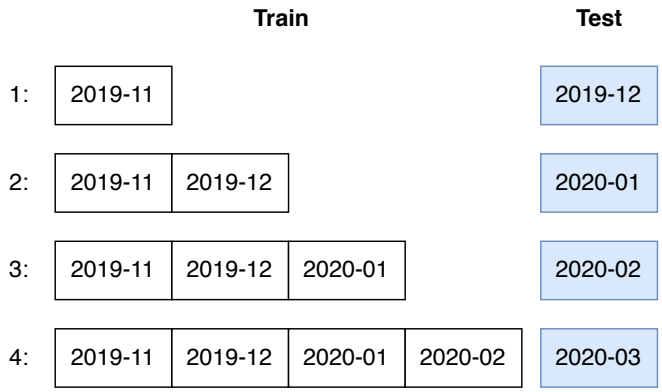


Figure 8 – Train and test sets for Setup 1.
(Source: the author.)

To apply the ARIMA model, *auto-arima* from `pmdarima` library[1] was used since it auto-tunes its hyperparameters. This approach was applied because the preparation of the hyperparameters for ARIMA is a complex and time-consuming task. The *auto-arima* technique performs several procedures automatically, making the process simpler and faster, finding the best hyperparameters for each data entry.

Table 4 presents the hyperparameters of LSTM and TCN. Both DL methods share some training hyperparameters. Adam was selected as optimizer, Mean Absolute Error (MAE) as loss function, 25 as the number of epochs and batch sizes of 32 and 128. Using MAE means to minimize the Manhattan distance (L_1 norm) of the differences between the predictions and the true output values.

For TCN, Rectified Linear Unit (ReLU) was adopted as activation function and each block output has a residual connection. For this method, `keras-tcn` library[2] (REMY, 2020) was adopted. The hyperparameters not specified previously assumed the default configuration of Keras-TensorFlow[3].

5.2 Setup 2 - Evaluating the Impact of Adversarial Attacks

In Setup 1, the objective is to find the models with the best predictive performance. In Setup 2, it is evaluated the impact of adversarial attacks against these models at test time. For this purpose, the attacker is considered to have limited computational and knowledge capabilities, as follows.

[1] <https://pypi.org/project/pmdarima/>
[2] <https://pypi.org/project/keras-tcn/>
[3] <https://www.tensorflow.org/api_docs/python/tf/keras/>

Table 4 – Configuration of hyperparameters for TCN and LSTM in Setup 1.

Method	Hyperparameter	Experimental choice
Both	Batch	32, 128
	Epochs	25
	Loss function	MAE
	Optimizer	Adam
LSTM	l	1, 2, 3
	Units	32, 64
	Dropout	0
TCN	k	2, 3
	b	1, 2
	Number of filters	32, 64
	Dropout rate	0
	d*	[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]

*The possible dilations followed Eq. 2.4.

5.2.1 Adversary's Goal

The adversary aims to carry out attacks at test time of the type indiscriminate, in the sense that the attacker modifies input data of all samples during operation to increase the prediction error of the victim's forecasting model F (induced during training time). In this case, F could be either TCN or LSTM.

5.2.2 Adversary's Knowledge

A gray-box attack (TABASSI et al., 2019) was experimented, in which the attacker has limited knowledge about training data and no knowledge about the model adopted by the victim. Particularly in our scenario, the attacker has access to the data collected during the first 2 months of operation. This decision was made to simulate a situation of weak information security during the first months of operation, making it easier for the attacker to access these data.

5.2.3 Adversary's Capability

The attacker acts during test time, modifying the input of test data only. To create the adversarial input, the attacker uses an adaptation of FGSM to the time series regression context, adding perturbations in the input of test data. As previously mentioned, the two main requirements of the perturbation are being not easily visually detected and, at the same time, degrading the performance of the victim's ML model.

Considering that the attacker performs a gray-box attack with no knowledge about F, he/she will rely on the transferability property, i.e., adversarial examples generated by the model F' can affect the performance of another model, trained using a different learning technique (CHAKRABORTY et al., 2018). In this work, it is assumed that the attacker also has limited computational resources, mimicking a scenario that the attacker uses a restricted device near to the plant for attacking it. This means that the attacker is not able to use complex substitute models to generate adversarial examples, so that simpler models should be adopted as F'.

For being simpler than DL, successful in classification scenarios (PAPERNOT; MCDANIEL; GOODFELLOW, 2016) and still differentiable, Logistic Regression (LR) was adopted to build the substitute model. Being differentiable makes easier to compute the gradient of the loss function, so that for LR (for further information, consult Appendix B) the gradient can be computed by Equation 5.1:

$$\nabla_x J = [\sigma(w.x + bias) - y]w. \tag{5.1}$$

This result enables crafting the perturbation component of FGSM. Figure 9 shows, in practical terms, how the available data is used to perform the attack.

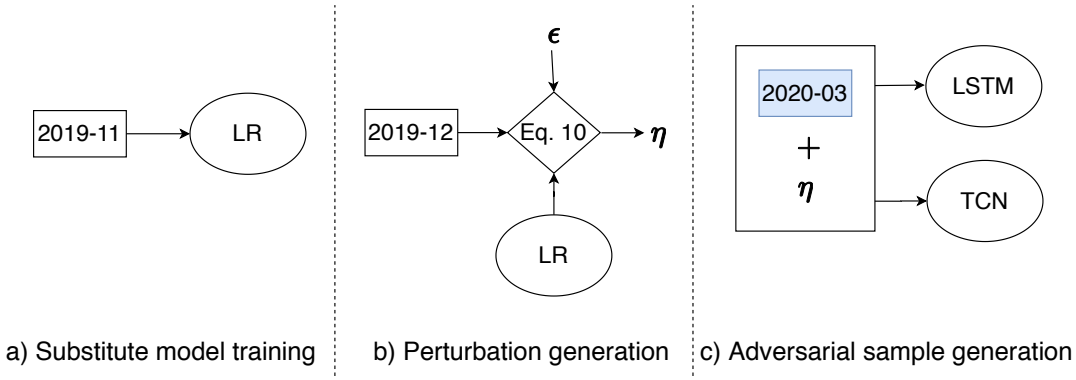


Figure 9 – Illustration of the adversary capability and knowledge.
(Source: the author.)

To obtain LR as F' (a), the first collected month is used as training data. To build the perturbation (b), along with the LR substitute model, the second month is used as $\mathbf{x}$ and y in Equation 2.13. Then, at test time (c), the adversarial sample x'_{test} for the corresponding legitimate test input x_{test} is computed by Equation 5.2:

$$x'_{test} = x_{test} + \eta \tag{5.2}$$

and inputted to the victim's model. The ϵ value was varied from 0.05 to 2 with steps of 0.05.

5.3 Setup 3 - Attack Detection and Defense

This setup was intended to evaluate the detection approach proposed in Chapter 4. In this experiment, we used an increased power generation data size of inverter 1. To train the forecasting models F, data from December 2019 to June 2020 was chosen. For testing, July and August. Information about them can be found in Table 5. Again, 20% was separated for the DL models hyperparameter tuning and the samples were collected every 15 minutes.

Table 5 – Information about train and test datasets in Setup 3.

Set	Maximum value [W]	Minimum value [W]	Average [W]	Number of samples
Train	43521.93	0	7283.43	20456
Test	34196.59	0	5635.50	5472

For the time series windowing, $i = 96$ was chosen. Besides, the forecast horizon $h = 1$ was taken. Table 6 shows the training hyperparameters for obtaining the forecasting model in Setup 3.

Table 6 – Configuration of hyperparameters for TCN and LSTM in Setup 3.

Method	Hyperparameter	Experimental choice
Both	Batch	32, 64, 128
	Epochs	25, 50
	Loss function	MAE
	Optimizer	Adam
LSTM	l	1, 2, 3
	Units	32, 64, 128
	Dropout	0
TCN	k	2, 3
	b	1, 2
	Number of filters	32, 64
	Dropout rate	0
	d*	[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]

*The possible dilations followed Eq. 2.4.

The defense approach also needed configurations. Minimum, Mean, Mean, Maximum, Standard deviation and Maximum used Python built-in functions. Ratio between Mean and Maximum, and Ratio between Minimum and Maximum were calculated based on those values. Entropy, Correlation, DFA and Hurst were obtained using `ndols` (SCHöLZEL, 2019)[4] Python module. These features made up the input of the detection classifiers.

[4] <https://nolds.readthedocs.io/en/latest/nolds.html#algorithms>

As for the training of OCC, the ν OCSVM hyperparameter alternated among 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4. For LOF, the contamination (fraction of outliers in the dataset) alternated between 0.00025 and 0.0005, and number of neighbors alternated among 20, 25, 30, 35, 40 and 45.

5.3.1 Threat Model

Attacking all the samples eases the detection of attackers by the system and operators, and also demands full capacity of inputting the perturbation generated by the attack algorithm. For these reasons, attacking specific samples is a more realistic assumption. In this setup, differently from Setup 2, where indiscriminate attack was adopted, a targeted attack was assumed as the adversary goal relatively to specificity.

Modeling the behavior of attackers is an intricating task and exhaustive options are possible. For practical purposes, we defined 3 different patterns that the attacks could adopt for selecting attacked instances:

1) Random: the attacked instances are chosen by the attacker at random. In this choice, the attack could be confused with the plant intrinsic noise (KHALYASMAA et al., 2020);

2) Intermittent (inter): every attacked instance is followed by a non-attacked instance, and vice-versa. In (KONTOURAS; TZES; DRITSAS, 2019), this pattern showed to be hardly detected by an estimation-based detector, mainly when the attack occurred at the same time of the signal change;

3) Sinusoidal (sin): a group of attacked instances is followed by a group of non-attacked instances, and vice-versa. This function takes as argument the instance index in radians. If the result is negative or zero, the instance is attacked. This pattern corresponds to a smoother version of intermittent attack in relation to frequency since in a fix period, the number of changes is lower in this pattern.

Again a gray box attack was assumed with limited knowledge about training data and no knowledge about the model adopted by the victim. The attacker will take 2019-12 to train the LR substitute model and 2020-01 for using with FGSM. As the test set was bigger than the size of 2020-01, it had to be used repeatedly for crafting the attack.

5.4 Evaluation Metrics

To compute the forecasting model performance, the Root Mean Squared Error (RMSE) was assessed in each test sets as:

$$RMSE = \sqrt{\frac{\sum_{j=1}^{n}(\hat{x}_{j+h} - x_{j+h})^2}{n}}, \tag{5.3}$$

where n corresponds to the number of samples of the test set. A characteristic of this error metric is robustness relatively to undesirable large deviations (AHMED et al., 2020).

For measuring the classification performance of the adversarial attacks detection models in Setup 3, F1-Score was chosen since it represents a balance between precision and recall. It can be expressed as (GOUTTE; GAUSSIER, 2005):

$$F1 - Score = \frac{2 \times precision \times recall}{precision + recall} \tag{5.4}$$

The non-parametric Friedman test (FRIEDMAN, 1937), with significance level at $\alpha = 0.05$ (FRIEDMAN, 1940), and the Nemenyi post-hoc test (NEMENYI, 1963) were adopted for statistical comparison. The former's null hypothesis is that all the compared groups are statistically equivalent at α. If this does not hold, the Nemenyi post-hoc can be applied, showing that there is a difference between two groups if their average ranks differ by a Critical Difference (CD) (DEMŠAR, 2006). These results can be observed in a CD diagram, in which in the top line the average rank of the different groups are plotted and groups within the same CD are connected.

5.5 Chapter Conclusion

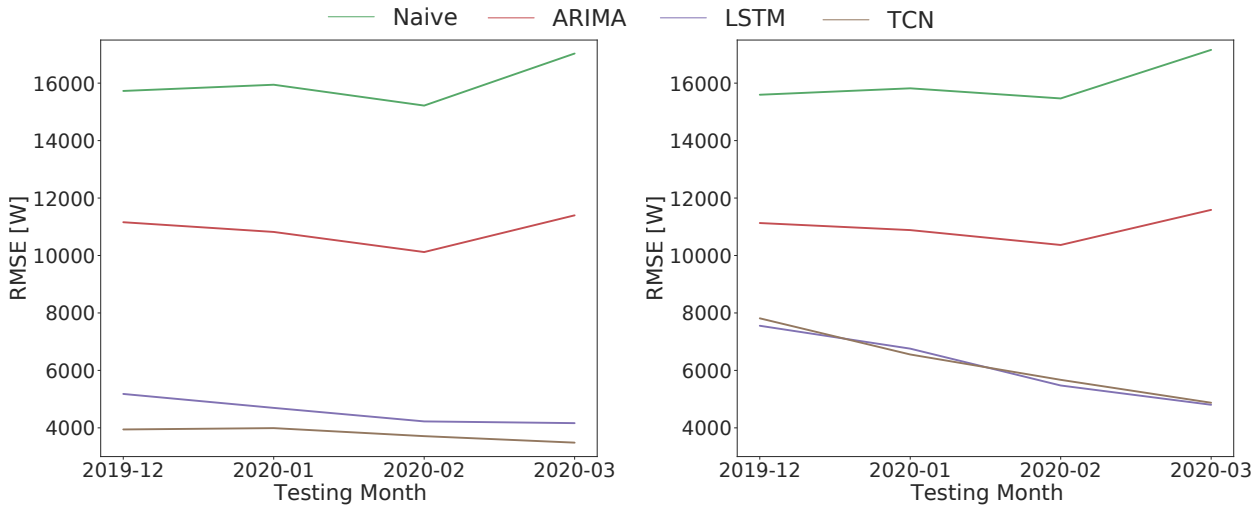
This chapter presented the studied PV power generation time series and the three main sets of experiments. The first is designed to assess the most promising forecasting models. The second is intended to verify their performance under adversarial attacks. The last to asses the defense approach against these attacks. In the next chapter, the results outputted by these experiments are registered and discussed.

6 RESULTS

This chapter shows the results of the defined experimental setups. It first compares the performance of the forecasting models. After that, it presents how the FGSM attack with different perturbation magnitudes influenced the performance of the best models. Finally, it shows the efficacy of the defense approach to distinguish between the legitimate and adversarial inputs and its impact on the forecasting model's performance.

6.1 Setup 1 - Obtaining the Best Forecasting Model

Initially, the forecasts of the four methods were compared. For this purpose, Figure 10 presents the test error obtained by the assessed models.



a) $h = 1$ b) $h = 96$

Figure 10 – RMSE of the methods using one month as testing period and the two experimented forecasting horizons. For LSTM and TCN, the mean values for the different hyperparameters configurations are shown.
(Source: the author.)

As observed, the Naive and ARIMA models had the worst performances, incurring an error of around 16 and 11 kW, respectively. These errors are greater than the average power generation value for the last test month (8506.43 W) and represent around 36% and 25% in relation to the maximum value.

Moreover, for the last test set, the error increased for both methods. During the operation of the plant, the derivatives between two consecutive samples were very high, which probably led to a worse performance due to the lack of ability of these two methods to anticipate fast changes.

For the DL models, the error tended to monotonously decrease as the amount of training data increased. The performance improvement is more evident for $h = 96$: the mean error decreased from almost 8 kW to nearly 5 kW.

As observed, the DL models obtained the best forecasting performance. To better analyze them, Table 7 shows the hyperparameters configurations of TCN and LSTM that led to the lowest forecast errors in each test set.

Table 7 – Best results of TCN and LSTM for both forecasts horizons and the test datasets. The bold values correspond to the best RMSE values for LSTM and TCN in each forecast horizon.

h	**Test**	**Method**	**RMSE [W]**	**Batch**	k	d	b	**Filters**	l	**Units**
1	2019-12	TCN	3625.71	32	2	[1, 4, 12, 48]	1	64	-	-
		LSTM	4276.75	32	-	-	-	-	1	32
	2020-01	TCN	3811.69	32	2	[1, 4, 12, 48]	1	64	-	-
		LSTM	4067.07	32	-	-	-	-	1	32
	2020-02	TCN	3576.19	32	2	[1, 2, 6, 12, 24]	2	64	-	-
		LSTM	3705.22	32	-	-	-	-	1	64
	2020-03	TCN	**3333.10**	128	2	[1, 3, 6, 12, 24]	2	32	-	-
		LSTM	**3672.89**	32	-	-	-	-	1	64
96	2019-12	TCN	7320.73	32	3	[1, 2, 4, 8, 16]	2	64	-	-
		LSTM	7294.97	128	-	-	-	-	2	32
	2020-01	TCN	6265.15	32	2	[1, 2, 6, 12, 24]	2	32	-	-
		LSTM	6016.31	128	-	-	-	-	1	32
	2020-02	TCN	5276.89	128	3	[1, 2, 4, 8, 16, 32]	1	32	-	-
		LSTM	5175.33	32	-	-	-	-	1	32
	2020-03	TCN	**4497.43**	128	2	[1, 4, 12, 48]	1	32	-	-
		LSTM	**4408.46**	128	-	-	-	-	1	32

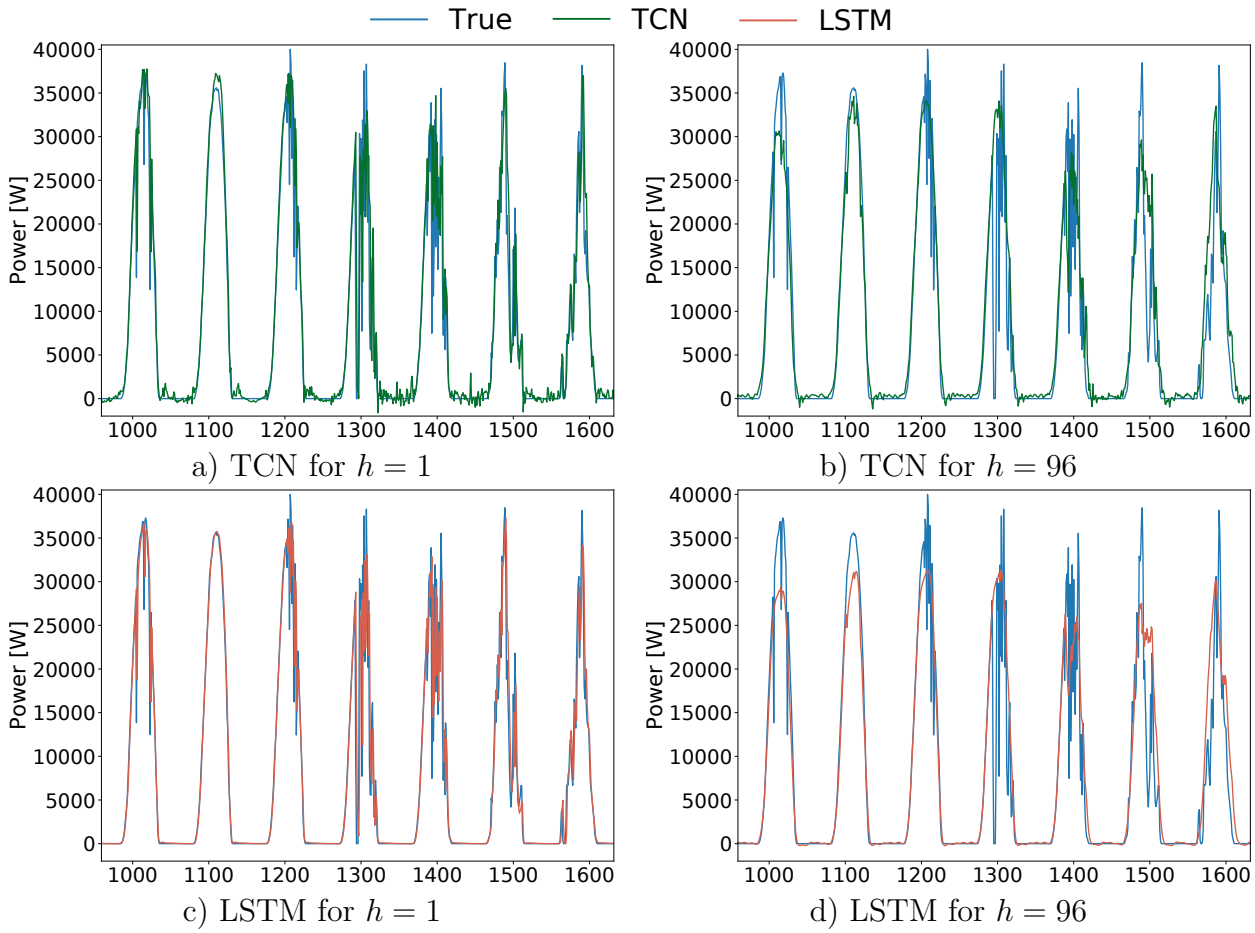
For $h = 1$, TCN achieved the best performance, with an error of 3333.10 W. It represented 29.24% and 19.57% in relation to ARIMA and Naive RMSEs for the same test set and 7.66% in relation to the maximum power value of 2020-03 (43521.93 W).

For $h = 96$, LSTM slightly outperformed TCN, with an error of 4408.56W. It represented 38.04% of the error presented by ARIMA, 25.69% in relation to Naive and 10.13% in relation to the maximum power value of 2020-03.

As to the hyperparameters, for $h = 1$, LSTM presented a preference for smaller batch size and a single layer (i.e, no stacking). As the training sample size increased, the number of required units also increased. For the same forecast horizon, the preferred kernel size of TCN was 2 and higher training sample sizes preferred more blocks. The last month preferred the largest batch size and a lower number of filters.

Still analyzing the hyperparameters, for $h = 96$, LSTM preferred 32 units for all training sample sizes and mostly only one layer. The last month preferred the largest batch size. TCN mostly required 32 filters, and as the training sample size increased, the batch size and the number of blocks increased as well.

Alongside the error values, we also evaluated qualitatively the models. With this intention, Figure 11 compares the true output values from 7 days of the last test set to the forecast results that led to the lowest RMSE for each h.



a) TCN for $h = 1$ b) TCN for $h = 96$

c) LSTM for $h = 1$ d) LSTM for $h = 96$

Figure 11 – Comparison between true output and predicted results by the best TCN and LSTM models.

(Source: the author.)

In general, the predictions followed the true output values. TCN presented lower stability around 0 than LSTM, particularly for $h = 1$. When $h = 96$ and the true output values are very high, it is observable a conservative forecast of the power generation.

In the figures we compare the same 7 days for both $h = 1$ and $h = 96$. The error in the predictions on the left ($h = 1$) and on the right ($h = 96$) are notably different, corroborating with the fact that forecasting models with $h = 1$ resulted in better performance than $h = 96$.

Since we evaluated the performance of the models with monthly updates, we statistically compared the RMSE obtained for all the models in each month. Figures 12 and 13 show the Nemenyi post-hoc test for $h = 1$ and $h = 96$, respectively.

As observed, the most recent test months obtained the first places in the rank, and the farthest months occupied the last places in the rank. In addition, the results of

Figure 12 – Nemenyi comparison of the testing months 2019-12, 2020-01, 2020-02 and 2020-03 for $h = 1$. The first in the rank corresponds to the lowest (best) RMSE.

(Source: the author.)

Figure 13 – Nemenyi comparison of the testing months 2019-12, 2020-01, 2020-02 and 2020-03 for $h = 96$. The first in the rank corresponds to the lowest (best) RMSE.

(Source: the author.)

these months were not connected by the CD, except for the months 2020-01 and 2019-12 when $h = 1$. This results ratifies that increasing the training size and updating the models might be beneficial for the plant.

6.2 Setup 2 - Evaluating the Impact of Adversarial Attacks

As previously mentioned, the adversarial input data must be subtle for not being easily visually identified. Considering this, Figure 14 compares one sample of an original input window that belongs to the test set and its respective sample of a maliciously crafted window input generated by FGSM.

Given the high similarity between the shape of both curves, it is very difficult to distinguish between the adversarial and the legitimate input, showing that a human would have difficulties to visually detect the attack. Although small, the difference in both curves increases as ϵ increases, and the most noticeable difference occurs when the curve starts to increase more intensively or stops decreasing.

To demonstrate how the adversary perturbation influenced the models' performance, Table 8 shows the error increase caused by the different ϵ values to the best DL models obtained in Setup 1 (when test set was 2020-03).

For $h = 1$ and $\epsilon = 0.05$, nearly no increase in error was found for both methods, but when ϵ slightly varied from 0.05 to 0.1, the error increase was multiplied by a factor of 5.17 for LSTM and 16.63 for TCN. The biggest error was found for LSTM when $\epsilon = 0.2$.

For $h = 96$, the biggest error increase was found for TCN, reaching almost 78%. Such error increase may cause serious harm to the grid operation. Suppose a situation

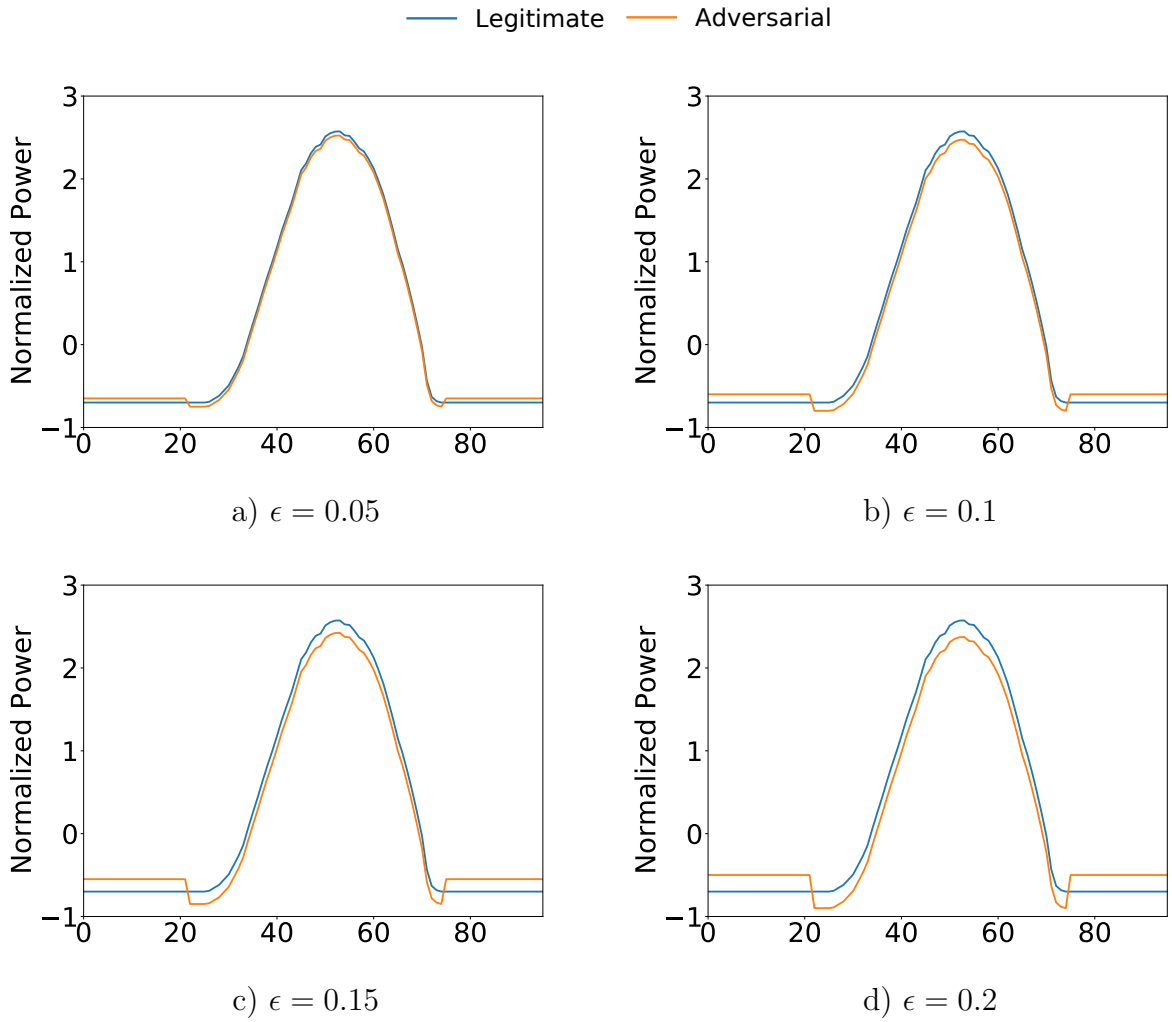


Figure 14 – Sample of legitimate input window and adversarial input window according to variation in ϵ following the procedure described in Setup 2.
(Source: the author.)

Table 8 – Percentual error increase in relation to the corresponding original dataset and method.

	LSTM				**TCN**			
	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.15$	$\epsilon = 0.2$	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.15$	$\epsilon = 0.2$
$h = 1$	0.88%	4.55%	10.75%	19.05%	0.22%	3.66%	9.97%	17.86%
$h = 96$	2.70%	17.77%	42.33%	57.06%	4.20%	21.03%	50.24%	77.99%

where the legitimate forecast, i.e., the TCN legitimate output, is 10000 W and there is a fixed demand of 20000 W for the region connected to the studied PV plant. Considering our most extreme setup for an attack ($\epsilon = 0.2$), the TCN output could be changed to 17799 W. Then, based on this wrong forecast, the system operator would assume that it would be necessary to deliver only more 2201 W from other plants to meet the demand of that region. However, during the operation, the operator would be actually required to deliver 10000 W to properly meet this demand. This difference would unbalance the grid and possibly culminate in a power outage.

On the other hand, if the wrong forecast leads to a generation value very bellow the real one, other power sources would be programmed to supply the required demand, minimizing the benefits that the PV plant can bring (e.g., saving costs and using a renewable power source). Besides, a grid unbalance is also expected.

We statistically compared the results of LSTM and TCN without being attacked (True_LSTM and True_TCN) with their corresponding versions under attack (Adversarial_LSTM and Adversarial_TCN). Figure 15 shows the CD diagram with this comparison.

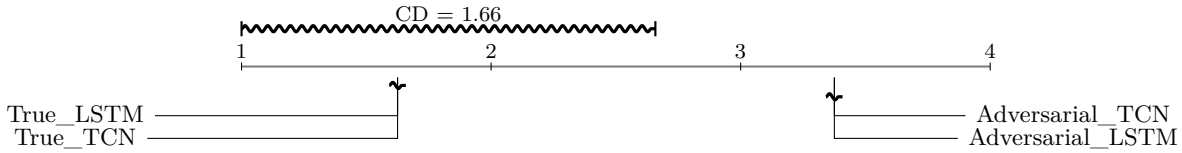


Figure 15 – CD diagram comparing the results of LSTM and TCN with and without attacks. The first in the rank corresponds to the lowest RMSE values. (Source: the author.)

These results show that LSTM and TCN had equivalent performance without attack; however, this performance was statistically different from TCN and LSTM under attack. This ratifies once again the degrading influence of adversarial examples. It is also noteworthy mentioning that TCN and LSTM were trained with data from four months, whereas LR was trained only with a month. Even so the damage in performance was significant.

Additionally, it is also possible to verify that FGSM was an efficient adversarial sample generator for the studied PV univariate time series regression, based only on a limited amount of historical data and using LR as a substitute model. As a consequence, LR was able to perform a cross-technique transferability to LSTM and TCN, since adversarial examples crafted for LR also affected TCN and LSTM. Thus, in the context of this work, a defense approach should be developed for the adopted forecasting models for preventing them from adversarial attacks.

6.3 Setup 3 - Attack Detection and Defense

Setup 3 considered a dataset with more collect historical data when compared to the previous setups. In this scenario, the lowest RMSE produced by LSTM and TCN were 1370.35 and 1388.24 W, respectively. These values corresponded to nearly 4% of the maximum power generation of the test period[1].

[1] For reference, in Setup 1, the TCN result for the test set composed of 2020-03 and $h = 1$ corresponded to 7.66% of the maximum power of that period

In this scenario, the best configuration for LSTM and TCN were batch sizes of 128 and number of epochs equal to 25. Besides, 3 stacked layers and 128 units for LSTM, and 32 filters, kernel size of 2, no staking and the dilation of [1, 2, 4, 8, 12, 24, 48] for TCN.

After obtaining the performance of both models, we executed the adversarial attacks against them and verified their forecasting results. In the sequence, we coupled the proposed defense approach and verified again the models' performance. These results are available in Table 9, which presents the lowest RMSE obtained using different combinations of attack magnitudes, patterns and classifiers. The error increases of TCN and LSTM under attack in relation to their performance without attack are also exhibited.

Table 9 – Comparison among RMSE obtained under attacks with and without defense mechanisms. The highest RMSE increase for each group was highlighted in bold.

ϵ	**Pattern**	**Detector**	**TCN RMSE [W]**	**TCN Increase [%]**	**LSTM RMSE [W]**	**LSTM Increase [%]**
-	-	-	1388.24	-	1370.35	-
0.05	Random	-	2190.24	57.77	2352.63	71.68
0.05	Intermittent	-	2459.39	77.16	2548.81	86.00
0.05	Sinusoidal	-	2583.73	86.12	2640.06	92.66
0.1	Random	-	3555.81	156.14	3053.74	122.84
0.1	Intermittent	-	4232.31	204.87	3506.88	155.91
0.1	Sinusoidal	-	4539.49	227.00	3740.08	172.93
0.15	Random	-	4944.43	256.17	4303.76	214.06
0.15	Intermittent	-	5977.85	330.61	5135.37	274.75
0.15	Sinusoidal	-	6399.79	361.00	5489.64	300.60
0.2	Random	-	6162.65	343.92	5739.74	318.85
0.2	Intermittent	-	7410.23	433.79	6878.50	401.95
0.2	Sinusoidal	-	7953.43	**472.91**	7388.21	**439.15**
0.05	Random	OCSVM	1973.35	42.15	2089.97	52.51
0.05	Intermittent	OCSVM	1922.94	38.52	2060.56	50.37
0.05	Sinusoidal	OCSVM	1933.91	39.31	2085.73	52.20
0.1	Random	OCSVM	1584.18	14.11	1769.21	29.11
0.1	Intermittent	OCSVM	1560.43	12.40	1725.28	25.90
0.1	Sinusoidal	OCSVM	1525.18	9.86	1701.40	24.16
0.15	Random	OCSVM	1740.24	25.36	1825.11	33.19
0.15	Intermittent	OCSVM	1675.32	20.68	1811.39	32.18
0.15	Sinusoidal	OCSVM	1618.52	16.59	1794.57	30.96
0.2	Random	OCSVM	2011.03	**44.86**	2115.47	**54.37**
0.2	Intermittent	OCSVM	1963.32	41.43	2104.91	53.60
0.2	Sinusoidal	OCSVM	1926.47	38.77	2041.83	49.00
0.05	Random	LOF	1881.34	**35.52**	2065.29	**50.71**
0.05	Intermittent	LOF	1880.88	35.49	2065.24	**50.71**
0.05	Sinusoidal	LOF	1880.88	35.49	2065.24	**50.71**
0.1	Random	LOF	1564.51	12.70	1872.29	36.63
0.1	Intermittent	LOF	1564.51	12.70	1871.65	36.58
0.1	Sinusoidal	LOF	1564.51	12.70	1871.60	36.58
0.15	Random	LOF	1497.77	7.89	1716.30	25.25
0.15	Intermittent	LOF	1497.77	7.89	1716.30	25.25
0.15	Sinusoidal	LOF	1497.77	7.89	1716.30	25.25
0.2	Random	LOF	1799.46	29.62	1971.70	43.88
0.2	Intermittent	LOF	1799.45	29.62	1971.69	43.88
0.2	Sinusoidal	LOF	1799.45	29.62	1971.70	43.88

Without attacks, LSTM provided lower RMSEs than TCN. Under attack and without defense, except for $\epsilon = 0.05$, LSTM still provided lower RMSEs than TCN. However, considering attacks using the proposed defense, TCN outperformed LSTM in

all scenarios. One reason for this is that LSTM is solidly grounded in the sequential information of the time series, and our mitigation uses the last valid prediction when under attack, displacing the sequence. In other words, this defense strategy better fits TCN due to the usage of local and global information of time series.

The greatest error increase in the table for TCN was 472.91% and occurred using sinusoidal attack pattern, $\epsilon = 0.2$ and no detector. For this same pattern and ϵ, using OCSVM as detector resulted in an RMSE increase of only 38.77%, and LOF as detector of 29.62%.

As for LSTM, the greatest error increase was 439.15% also with sinusoidal attack pattern and $\epsilon = 0.2$. Using this same configurations but now defending the system, this RMSE increase was considerably lower: 49.00% for OCSVM and 43.88% for LOF.

It is also noteworthy mentioning that when $\epsilon = 0.05$, the differences between adversarial and legitimate samples are harder to distinguish, which hinders the error reduction by OCSVM and LOF. These models achieved the lowest RMSE facing the ϵ values of 0.1 and 0.15, reaching error increases of only 9.86% and 7.89%. When $\epsilon = 0.2$, the difference between these kind of samples is more evident, which eases the detection and could induce to assume that the error increase would be the lowest for this ϵ. However, when the classifier fails to detect an adversarial sample, the error caused by this ϵ is more influential than lower attack magnitudes.

Still related to this point, without a detector the $\epsilon = 0.2$ was clearly more advantageous for the attacker and $\epsilon = 0.05$ represented the most modest error increase. Nevertheless, when using LOF, the attacker would benefit more from using the lowest ϵ. Remarkably, even in the best attack scenario using the detection approach, the highest error increases are lower than the lowest error increase without detectors, which supports once again the use of detectors.

Three different attack patterns (sinusoidal, intermittent and random) were employed when causing the degradation. Figure 16 compares the rank and CD for them. All the RMSEs of TCN and LSTM for the attacks without detector and for all classifiers hyperparameters were considered.

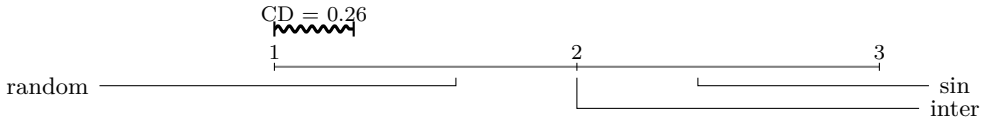


Figure 16 – CD diagram comparing the sinusoidal, intermittent and random attack patterns. The first in the rank corresponds to the greatest RMSE values (best for the attacker).

(Source: the author.)

As observed, random attack pattern was the one that brought more negative

results for the forecasting models, followed by intermittent and sinusoidal. It is interesting that all these attacks produces roughly 50% of legitimate samples and 50% of adversarial samples. Though, their performances are not statistically equivalent, suggesting that the frequency the attack occurs or the time the adversarial samples are generated also exert influence in the deterioration magnitude.

Regarding the detectors used in the proposed approach, different hyperparameters were experimented to find the most suitable detector for adversarial attacks. Now we evaluate their detection performance. Figure 17 shows different F1-Score obtained by varying the Number of Neighbors and Contamination across different attacks and ϵ for LOF.

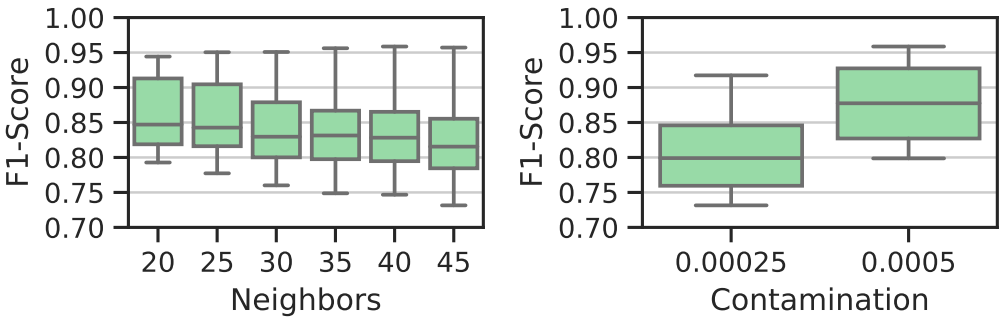


Figure 17 – F1-Score obtained with LOF across several Number of Neighbors and Contamination hyperparameters.
(Source: the author.)

Lower number of neighbors delivered slightly better medians of the boxplots. The lowest number of neighbors, 20, delivered the best average F1-Score of 86.05%. For Contamination, the best general performances were obtained with the highest value 0.0005, reaching an average F1-Score of 87.94%.

Figure 18 compares F1-Score for OCSVM using different values for ν. Lower values of this hyperparameter were advantageous for this classifier. It is also shown a slightly difference in terms of performance for 0.1 and 0.15, ν=0.1 reached an F1-Score of 93.12% for detecting sin adversarial attack function.

Previously we analyzed the different attack patterns in terms of RMSE of the forecasting model. In Figure 19 we analyze them in terms of the detection F1-Score using LOF and OCSVM.

OCSVM achieved higher median F1-Score than LOF for the three attack patterns. Actually, the median of OCSVM was very close to the 3^{rd} quartile of LOF and the minimum values of OCSVM were very close to the median of LOF. The three patterns showed very similar box, indicating that similar F1-Scores of the detectors can result in different performance of the forecasting models.

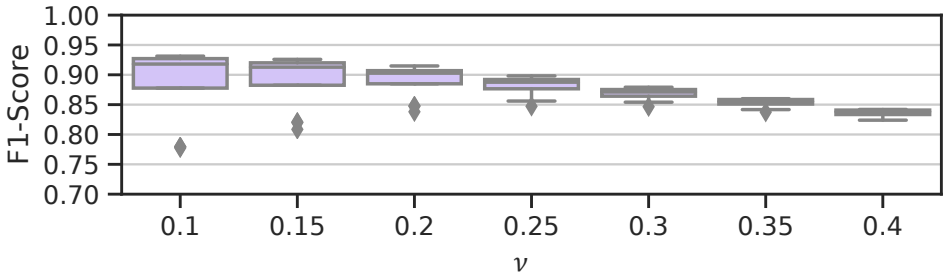


Figure 18 – F1-Score obtained with OCSVM using seven ν values.
(Source: the author.)

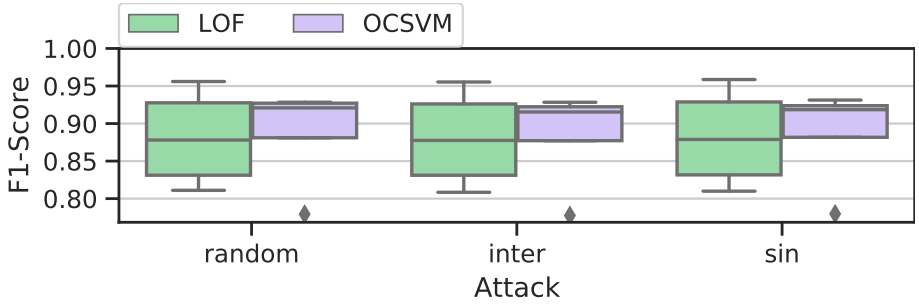


Figure 19 – F1-Score variation of LOF and OCSVM grouped by attack perturbation function.
(Source: the author.)

The features used in the approach were promised to have low computational calculation cost. To verify that, we also calculated the computation time of the features in Table 10.

Table 10 – Average computation time of the features extracted for each input window.

Feature	**Computation Time [seconds]**
Min	0.0003 ± 0.0001
Mean	0.00029±0.00004
Median	0.00027±0.00004
Max	0.00026±0.00004
Mean to Max Ratio	0.00032±0.00004
Min to Max Ratio	0.00031±0.00005
STD	0.00028±0.00004
Entropy	0.0035±0.0008
Correlation	0.005±0.004
DFA	0.015±0.001
Hurst	0.005±0.002
Total	0.0297±0.008

For each input window, the total average time was 0.0297 seconds with deviation of 0.008. DFA showed to be the most costly feature to be calculated, taking in average 0.015 seconds with an standard deviation of 0.004. On the other hand, the Maximum

feature was the most lightweight. As the sampling interval for acquiring the data is 15 minutes, this computation do not overload the functioning of the plant.

6.4 Chapter Conclusion

First, it was observed that TCN and LSTM presented outstanding results as forecasting models. After that, it was presented how FGSM was able to degrade their performances in a constrained knowledge and capability scenario. Lastly, the use of the proposed defense approach showed that the deterioration in performance of the forecasting models caused by adversarial attacks can be greatly reduced. In the next chapter, the conclusions of this master's research and future work are exposed.

7 CONCLUSION

This work had three main goals: obtaining forecasting models for PV power generation, investigating the impact of adversarial attacks during test time and proposing a defense approach against this kind of attack.

Regarding the evaluation of the forecasting models, results showed that DL models outperformed Naive and ARIMA. Those models were more able to deal with the high complexity involving PV generation data. Besides, the smaller the forecast horizon, the better the performance. It is understandable, since forecasting what is going to happen after 15 minutes is easier than forecasting 24 hours ahead.

Moreover, it was observed that the bigger the training sample size, the better the TCN and LSTM performances. This implies that updating these models according to the functioning of the plant can bring accuracy improvements. Considering that the RMSE obtained by TCN and LSTM represented around 8% of maximum power of the last test period (Setup 1) when $h = 1$ and 10% when $h = 96$, these models were considered satisfactory.

As for the attacks at test time, FGSM was increased models' error even with small ϵ values and limited knowledge about the target. The error increase for $h = 96$ was greater than for $h = 1$, achieving up to 77.99% for TCN and $\epsilon = 0.2$.

It was also possible to validate the cross-technique transferability of LR as a substitute model for generating adversarial examples for LSTM and TCN. The fact of LR being simpler yet effective eases generating adversarial examples in computational limited hardware, which is a more realistic assumption about the attacker.

Additionally, the proposed detection approach using LOF and OCSVM was able to defend the forecasting models against the adversarial attacks, reaching a maximum forecasting error increase of 55% against a maximum of 472.91% of error increase without any defense approach. Besides, it was able to achieve an error increase of only 7.89% for $\epsilon = 0.15$ and LOF.

In relation to the attack patterns, random pattern significantly increased the RMSE when compared to intermittent and sinusoidal. However, the detectors performances did not vary greatly according to the different patterns.

With respect to the classifiers hyperparameters, OCSVM delivered best F1-Score results for $\nu = 0.1$. For LOF, the lowest number of neighbors and highest contamination were preferred.

These results showed that the proposed approach was effective for defending the studied forecasting models. Both OCSVM and LOF were able to substantially alleviate the effect of the adversarial attacks when used as detectors, and the features extraction did not incur in excessive computational costs.

7.1 Open Issues and Future Works

As more data become available, it can be used to improve the forecasting results. Using a growing input window of historical data to train the forecasting model has a drawback of increasing the complexity for updating the models. Then, better exploring the properties of this time series, such as stationarity, and investigating ways of selecting its most representative data is a possible research path.

Further, treating the time series as multivariate by considering exogenous related variables could improve the models and aid in anticipating events. For instance, irradiation and temperature are highly influencing factor for this type of plant and could improve the performance of the methods.

In this work we assumed limited knowledge of the attacker in relation to the model and limited computational power for computing the adversarial samples. Exploring variations in the level of knowledge and other substitute models are possibilities for future research.

As a foundation work, we chose FGSM by its importance in adversarial ML literature. However, the development/adaptation of other attack algorithms for regression problems is encouraged. It is remarkable mentioning that the lower the knowledge about the the possible attacks, the easier for the attackers to unveil the vulnerabilities about the forecasting models. Thus, the broader the studies on the risks, the greater the chances of success of a proactive approach.

Closely related to that, we simplified the attacker behavior to random, intermittent and sinusoidal. However, the attacker might adopt more sophisticated attack patterns, and capturing them can be advantageous for further prevention.

Besides, other detectors and defense mechanisms can be adopted. In this work we opted for replacing an attacked sample by the last predicted normal sample, but only removing it, for instance, would be possible.

All in all, there is still numerous research branches to be expanded in the context of PV generation and the security aspects related to adversarial attacks to this domain. We hope the results obtained in this work can be an impulse for future works in the area, preventing the negative impacts that adversarial attacks to PV generation are prone to cause.

BIBLIOGRAPHY

ABDU-AGUYE, M. G. et al. Detecting adversarial attacks in time-series data. In: IEEE. *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.], 2020. p. 3092–3096.

AHMED, R. et al. A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 124, p. 109792, 2020.

AKHTAR, N.; MIAN, A. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, IEEE, v. 6, p. 14410–14430, 2018.

AKHTER, M. N. et al. Review on forecasting of photovoltaic power generation based on machine learning and metaheuristic techniques. *IET Renewable Power Generation*, IET, v. 13, n. 7, p. 1009–1023, 2019.

ALFELD, S.; ZHU, X.; BARFORD, P. Data poisoning attacks against autoregressive models. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2016. v. 30, n. 1.

ANEEL. *Brasil alcança 170 mil megawatts de capacidade instalada em 2019*. 2019. Disponível em: <http://bit.ly/35IP2Vo>.

ANTONANZAS, J. et al. Review of photovoltaic power forecasting. *Solar Energy*, Elsevier, v. 136, p. 78–111, 2016.

ATIQUE, S. et al. Forecasting of total daily solar energy generation using ARIMA: A case study. In: IEEE. *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. [S.l.], 2019. p. 0114–0119.

BAI, S.; KOLTER, J. Z.; KOLTUN, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

BIGGIO, B. et al. Evasion attacks against machine learning at test time. In: SPRINGER. *Joint European conference on machine learning and knowledge discovery in databases*. [S.l.], 2013. p. 387–402.

BIGGIO, B.; ROLI, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, Elsevier, v. 84, p. 317–331, 2018.

BOMFIM, T. S. Evolution of machine learning in smart grids. In: IEEE. *2020 IEEE 8th International Conference on Smart Energy Grid Engineering (SEGE)*. [S.l.], 2020. p. 82–87.

BOX, G. E. P.; JENKINS, G. *Time Series Analysis, Forecasting and Control*. USA: Holden-Day, Inc., 1990. ISBN 0816211043.

BREUNIG, M. M. et al. Lof: identifying density-based local outliers. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. [S.l.: s.n.], 2000. p. 93–104.

BULUSU, S. et al. Anomalous instance detection in deep learning: A survey. *arXiv preprint arXiv:2003.06979*, 2020.

CERQUEIRA, V.; TORGO, L.; MOZETIČ, I. Evaluating time series forecasting models: An empirical study on performance estimation methods. *Machine Learning*, Springer, v. 109, n. 11, p. 1997–2028, 2020.

CERQUEIRA, V.; TORGO, L.; SOARES, C. Machine learning vs statistical methods for time series forecasting: Size matters. *arXiv preprint arXiv:1909.13316*, 2019.

CHAKRABORTY, A. et al. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

CHEN, Y.; TAN, Y.; DEKA, D. Is machine learning in power systems vulnerable? In: IEEE. *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. [S.l.], 2018. p. 1–6.

CHEN, Y.; TAN, Y.; ZHANG, B. Exploiting vulnerabilities of load forecasting through adversarial attacks. In: *Proceedings of the Tenth ACM International Conference on Future Energy Systems*. [S.l.: s.n.], 2019. p. 1–11.

DAS, U. K. et al. Forecasting of photovoltaic power generation and model optimization: A review. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 81, p. 912–928, 2018.

DAUT, M. A. M. et al. Building electrical energy consumption forecasting analysis using conventional and artificial intelligence methods: A review. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 70, p. 1108–1118, 2017.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, v. 7, n. Jan, p. 1–30, 2006.

DHILLON, G. S. et al. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

DIAMANTOULAKIS, P. D.; KAPINAS, V. M.; KARAGIANNIDIS, G. K. Big data analytics for dynamic energy management in smart grids. *Big Data Research*, v. 2, n. 3, p. 94 – 101, 2015. ISSN 2214-5796.

DOLARA, A. et al. A physical hybrid artificial neural network for short term forecasting of PV plant power output. *Energies*, Multidisciplinary Digital Publishing Institute, v. 8, n. 2, p. 1138–1153, 2015.

DOLARA, A.; LEVA, S.; MANZOLINI, G. Comparison of different physical models for PV power output prediction. *Solar Energy*, Elsevier, v. 119, p. 83–99, 2015.

DONG, Y. et al. Boosting adversarial attacks with momentum. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2018. p. 9185–9193.

DUCHESNE, L.; KARANGELOS, E.; WEHENKEL, L. Recent developments in machine learning for energy systems reliability management. *Proceedings of the IEEE*, IEEE, 2020.

ENNS, R. H. *It's a nonlinear world*. [S.l.]: Springer Science & Business Media, 2010.

FAWAZ, H. I. et al. Adversarial attacks on deep neural networks for time series classification. In: IEEE. *2019 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2019. p. 1–8.

FEINMAN, R. et al. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, Taylor & Francis, v. 32, n. 200, p. 675–701, 1937.

FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, JSTOR, v. 11, n. 1, p. 86–92, 1940.

GOODFELLOW, I. J.; SHLENS, J.; SZEGEDY, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

GOUTTE, C.; GAUSSIER, E. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: SPRINGER. *European conference on information retrieval*. [S.l.], 2005. p. 345–359.

GROSSE, K. et al. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

HAO-CHEN, H. X. Y. M. et al. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, Springer, v. 17, n. 2, p. 151–178, 2020.

HICKMAN, W. B. *The term structure of interest rates: an exploratory analysis*. [S.l.]: National Bureau of Economic Research, Financial Research Program, 1942.

HINTON, G.; VINYALS, O.; DEAN, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

HO, S.; XIE, M. The use of ARIMA models for reliability forecasting and analysis. *Computers & Industrial Engineering*, v. 35, n. 1, p. 213 – 216, 1998. ISSN 0360-8352.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HODGE, B.-M. et al. The combined value of wind and solar power forecasting improvements and electricity storage. *Applied Energy*, Elsevier, v. 214, p. 1–15, 2018.

ISLAM, S. N.; BAIG, Z.; ZEADALLY, S. Physical layer security for the smart grid: vulnerabilities, threats, and countermeasures. *IEEE Transactions on Industrial Informatics*, IEEE, v. 15, n. 12, p. 6522–6530, 2019.

JAGIELSKI, M. et al. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In: IEEE. *2018 IEEE Symposium on Security and Privacy (SP)*. [S.l.], 2018. p. 19–35.

JAIHUNI, M. et al. A partially amended hybrid bi-GRU—ARIMA model (PAHM) for predicting solar irradiance in short and very-short terms. *Energies*, Multidisciplinary Digital Publishing Institute, v. 13, n. 2, p. 435, 2020.

KARIM, F.; MAJUMDAR, S.; DARABI, H. Adversarial attacks on time series. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, 2020.

KHALYASMAA, A. I. et al. Industry experience of developing day-ahead photovoltaic plant forecasting system based on machine learning. *Remote Sensing*, Multidisciplinary Digital Publishing Institute, v. 12, n. 20, p. 3420, 2020.

KHAN, S. S.; MADDEN, M. G. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, Cambridge University Press, v. 29, n. 3, p. 345–374, 2014.

KHASHEI, M.; BIJARI, M. A novel hybridization of artificial neural networks and arima models for time series forecasting. *Applied Soft Computing*, Elsevier, v. 11, n. 2, p. 2664–2675, 2011.

KONTOURAS, E.; TZES, A.; DRITSAS, L. Hybrid detection of intermittent cyber-attacks in networked power systems. *Energies*, Multidisciplinary Digital Publishing Institute, v. 12, n. 24, p. 4625, 2019.

KOTU, V.; DESHPANDE, B. Chapter 12 - Time series forecasting. In: KOTU, V.; DESHPANDE, B. (Ed.). *Data Science*. Second edition. [S.l.]: Morgan Kaufmann, 2019. p. 395 – 445. ISBN 978-0-12-814761-0.

KURAKIN, A.; GOODFELLOW, I.; BENGIO, S. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

LARA-BENÍTEZ, P. et al. Temporal convolutional networks applied to energy-related time series forecasting. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 10, n. 7, p. 2322, 2020.

LEA, C. et al. Temporal convolutional networks for action segmentation and detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2017. p. 156–165.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LIU, X. et al. Towards robust neural networks via random self-ensemble. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. [S.l.: s.n.], 2018. p. 369–385.

LU, J.; ISSARANON, T.; FORSYTH, D. Safetynet: Detecting and rejecting adversarial examples robustly. In: *Proceedings of the IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2017. p. 446–454.

MADRY, A. et al. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

MAHMUD, K. et al. The impact of prediction errors in the domestic peak power demand management. *IEEE Transactions on Industrial Informatics*, IEEE, v. 16, n. 7, p. 4567–4579, 2019.

MCLAUGHLIN, R. L. Forecasting models: Sophisticated or naive? *Journal of Forecasting (pre-1986)*, Wiley Periodicals Inc., v. 2, n. 3, p. 274, 1983.

MEHRDAD, S. et al. Cyber-physical resilience of electrical power systems against malicious attacks: A review. *Current Sustainable/Renewable Energy Reports*, Springer, v. 5, n. 1, p. 14–22, 2018.

MELLIT, A. et al. Advanced methods for photovoltaic output power forecasting: A review. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 10, n. 2, p. 487, 2020.

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*. [S.l.: s.n.], 2017. p. 135–147.

METZEN, J. H. et al. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

MODHA, D. S.; MASRY, E. Prequential and cross-validated regression estimation. *Machine Learning*, Springer, v. 33, n. 1, p. 5–39, 1998.

NEMENYI, P. Distribution-free multiple comparisons (Doctoral dissertation, Princeton University, 1963). *Dissertation Abstracts International*, v. 25, n. 2, p. 1233, 1963.

NIAZAZARI, I.; LIVANI, H. Attack on grid event cause analysis: An adversarial machine learning approach. In: IEEE. *2020 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. [S.l.], 2020. p. 1–5.

NOBRE, A. M. et al. PV power conversion and short-term forecasting in a tropical, densely-built environment in singapore. *Renewable Energy*, Elsevier, v. 94, p. 496–509, 2016.

PAPERNOT, N.; MCDANIEL, P. Extending defensive distillation. *arXiv preprint arXiv:1705.05264*, 2017.

PAPERNOT, N.; MCDANIEL, P.; GOODFELLOW, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

PAPERNOT, N. et al. Sok: Security and privacy in machine learning. In: IEEE. *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. [S.l.], 2018. p. 399–414.

PAPERNOT, N. et al. Distillation as a defense to adversarial perturbations against deep neural networks. In: IEEE. *2016 IEEE Symposium on Security and Privacy (SP)*. [S.l.], 2016. p. 582–597.

PARZEN, E. et al. An approach to time series analysis. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 32, n. 4, p. 951–989, 1961.

PENA, E. H. et al. Anomaly detection using digital signature of network segment with adaptive ARIMA model and paraconsistent logic. In: IEEE. *2014 IEEE Symposium on Computers and Communications (ISCC)*. [S.l.], 2014. p. 1–6.

REMY, P. *Temporal Convolutional Networks for Keras*. [S.l.]: GitHub, 2020. <https://github.com/philipperemy/keras-tcn>.

ROUANI, B. D. et al. Safe machine learning and defeating adversarial attacks. *IEEE Security & Privacy*, IEEE, v. 17, n. 2, p. 31–38, 2019.

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*. [S.l.: s.n.], 2016. p. 25–32.

SANTANA, E. J. et al. Photovoltaic generation forecast: Model training and adversarial attack aspects. In: SPRINGER. *Brazilian Conference on Intelligent Systems*. [S.l.], 2020. p. 634–649.

SCHÖLKOPF, B. et al. Estimating the support of a high-dimensional distribution. *Neural computation*, MIT Press, v. 13, n. 7, p. 1443–1471, 2001.

SCHöLZEL, C. *Nonlinear measures for dynamical systems*. [S.l.]: Zenodo, 2019.

SHAO, X. et al. Domain fusion CNN-LSTM for short-term power consumption forecasting. *IEEE Access*, IEEE, v. 8, p. 188352–188362, 2020.

SUCIU, O. et al. When does machine learning FAIL? Generalized transferability for evasion and poisoning attacks. In: *27th USENIX Security Symposium (USENIX Security 18)*. [S.l.: s.n.], 2018. p. 1299–1316.

SZEGEDY, C. et al. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

TABASSI, E. et al. A taxonomy and terminology of adversarial machine learning. *NIST IR*, 2019.

TIAN, J. et al. Adaptive normalized attacks for learning adversarial attacks and defenses in power systems. In: IEEE. *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. [S.l.], 2019. p. 1–6.

TORRES, J. F. et al. Deep learning for big data time series forecasting applied to solar power. In: SPRINGER. *The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications*. [S.l.], 2018. p. 123–133.

TRAMÈR, F. et al. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

WANG, G.; SU, Y.; SHU, L. One-day-ahead daily power forecasting of photovoltaic systems based on partial functional linear regression models. *Renewable Energy*, Elsevier, v. 96, p. 469–478, 2016.

WANG, H. et al. A review of deep learning for renewable energy forecasting. *Energy Conversion and Management*, Elsevier, v. 198, p. 111799, 2019.

WANG, K.; QI, X.; LIU, H. A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network. *Applied Energy*, Elsevier, v. 251, p. 113315, 2019.

WANG, X. et al. The security of machine learning in an adversarial setting: A survey. *Journal of Parallel and Distributed Computing*, Elsevier, v. 130, p. 12–23, 2019.

XIE, C. et al. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

XU, W.; EVANS, D.; QI, Y. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

YANG, D. On post-processing day-ahead nwp forecasts using kalman filtering. *Solar Energy*, Elsevier, v. 182, p. 179–181, 2019.

YEN, C.-F. et al. Predicting solar performance ratio based on encoder-decoder neural network model. In: IEEE. *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. [S.l.], 2019. p. 1–4.

YUAN, X. et al. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, v. 30, n. 9, p. 2805–2824, 2019.

ZARPELÃO, B. B. et al. How machine learning can support cyberattack detection in smart grids. In: *Artificial Intelligence Techniques for a Scalable Energy Transition*. [S.l.]: Springer, 2020. p. 225–258.

ZHANG, D.; HAN, X.; DENG, C. Review on the research and practice of deep learning and reinforcement learning in smart grids. *CSEE Journal of Power and Energy Systems*, CSEE, v. 4, n. 3, p. 362–370, 2018.

Appendix

APPENDIX A – EXTRA RESULTS TABLES

Setup 1 involved hyperparameter tuning of the DL models. Tables 11 and 12 register the results of all experimented hyperparameters for TCN and LSTM models for the last test set.

Table 11 – Results for TCN and LSTM for the different hyperparameter configurations when $h = 1$.

Method	RMSE_train	RMSE_test	Batch	k	d	b	Filters	l	Units
LSTM	3877.71	4066.99	32					1	32
LSTM	5172.98	4582.67	32					1	32
LSTM	3853.69	3672.89	32					1	64
LSTM	5456.27	4949.91	32					1	64
LSTM	4279.25	4264.62	32					2	32
LSTM	5292.85	4984.04	32					2	32
LSTM	3929.79	3833.94	32					2	64
LSTM	5738.95	5064.33	32					2	64
LSTM	5079.18	5141.33	32					3	32
LSTM	5861.93	5423.91	32					3	32
LSTM	4408.56	4369.82	32					3	64
LSTM	5883.98	5252.47	32					3	64
TCN	3573.98	3455.16	32	2	[1, 4, 12, 48]	1	32		
TCN	5611.99	4765.56	32	2	[1, 4, 12, 48]	1	32		
TCN	3631.37	3420.26	32	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	5358.19	4802.56	32	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	3778.72	3596.84	32	2	[1, 3, 6, 12, 24]	2	32		
TCN	5565.97	5410.53	32	2	[1, 3, 6, 12, 24]	2	32		
TCN	3662.52	3385.94	32	2	[1, 2, 6, 12, 24]	2	32		
TCN	5250.25	4548.51	32	2	[1, 2, 6, 12, 24]	2	32		
TCN	4084.47	4138.86	32	3	[1, 4, 16, 32]	1	32		
TCN	5435.08	4885.83	32	3	[1, 4, 16, 32]	1	32		
TCN	3727.74	3519.96	32	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	5327.74	5016.49	32	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	3667.05	3542.69	32	3	[1, 2, 4, 8, 16]	2	32		
TCN	5307.14	4942.50	32	3	[1, 2, 4, 8, 16]	2	32		
TCN	3763.31	3540.13	32	3	[1, 4, 16]	2	32		
TCN	5383.83	5006.96	32	3	[1, 4, 16]	2	32		
TCN	3635.52	3488.49	32	2	[1, 4, 12, 48]	1	64		
TCN	5692.10	4884.34	32	2	[1, 4, 12, 48]	1	64		
TCN	3592.46	3346.45	32	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	5758.89	5061.86	32	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	3624.47	3401.12	32	2	[1, 3, 6, 12, 24]	2	64		
TCN	5877.02	5484.75	32	2	[1, 3, 6, 12, 24]	2	64		
TCN	3619.55	3363.92	32	2	[1, 2, 6, 12, 24]	2	64		
TCN	5662.96	5432.22	32	2	[1, 2, 6, 12, 24]	2	64		
TCN	3963.92	3802.00	32	3	[1, 4, 16, 32]	1	64		
TCN	5778.44	5494.21	32	3	[1, 4, 16, 32]	1	64		
TCN	3982.00	3457.77	32	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	6025.36	5464.23	32	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	3735.19	3496.61	32	3	[1, 2, 4, 8, 16]	2	64		
TCN	5911.72	5257.16	32	3	[1, 2, 4, 8, 16]	2	64		
TCN	3725.60	3402.52	32	3	[1, 4, 16]	2	64		
TCN	5608.26	4894.99	32	3	[1, 4, 16]	2	64		
LSTM	4161.84	3963.22	128					1	32
LSTM	5345.83	4408.46	128					1	32
LSTM	3985.70	3875.17	128					1	64
LSTM	5273.64	4582.88	128					1	64
LSTM	4422.43	4252.99	128					2	32
LSTM	5256.39	4609.41	128					2	32
LSTM	4240.04	3877.81	128					2	64
LSTM	5384.12	4477.65	128					2	64
LSTM	4522.80	4336.65	128					3	32
LSTM	5252.03	4518.08	128					3	32
LSTM	4376.59	4316.72	128					3	64
LSTM	5351.07	4786.32	128					3	64
TCN	3679.93	3432.31	128	2	[1, 4, 12, 48]	1	32		
TCN	5314.93	4497.43	128	2	[1, 4, 12, 48]	1	32		
TCN	3637.41	3402.64	128	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	5438.66	4622.66	128	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	3596.91	3333.10	128	2	[1, 3, 6, 12, 24]	2	32		
TCN	5421.31	4703.15	128	2	[1, 3, 6, 12, 24]	2	32		
TCN	3769.74	3483.55	128	2	[1, 2, 6, 12, 24]	2	32		
TCN	5504.00	4819.88	128	2	[1, 2, 6, 12, 24]	2	32		
TCN	3781.45	3500.03	128	3	[1, 4, 16, 32]	1	32		
TCN	5424.62	4519.38	128	3	[1, 4, 16, 32]	1	32		
TCN	3756.39	3523.69	128	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	5238.95	4718.58	128	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	3660.46	3436.77	128	3	[1, 2, 4, 8, 16]	2	32		
TCN	5302.32	4677.51	128	3	[1, 2, 4, 8, 16]	2	32		
TCN	3598.45	3351.58	128	3	[1, 4, 16]	2	32		
TCN	5426.24	4836.57	128	3	[1, 4, 16]	2	32		
TCN	3588.20	3379.56	128	2	[1, 4, 12, 48]	1	64		
TCN	5379.22	4635.63	128	2	[1, 4, 12, 48]	1	64		
TCN	3654.18	3423.69	128	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	5271.22	4536.82	128	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	3672.72	3403.19	128	2	[1, 3, 6, 12, 24]	2	64		
TCN	5470.24	4829.98	128	2	[1, 3, 6, 12, 24]	2	64		
TCN	3709.85	3509.94	128	2	[1, 2, 6, 12, 24]	2	64		
TCN	5393.45	4676.39	128	2	[1, 2, 6, 12, 24]	2	64		
TCN	3710.02	3492.49	128	3	[1, 4, 16, 32]	1	64		
TCN	5226.96	4506.82	128	3	[1, 4, 16, 32]	1	64		
TCN	3654.12	3519.65	128	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	5235.32	4590.65	128	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	3773.80	3515.91	128	3	[1, 2, 4, 8, 16]	2	64		
TCN	5379.21	4806.03	128	3	[1, 2, 4, 8, 16]	2	64		
TCN	3695.33	3461.47	128	3	[1, 4, 16]	2	64		

Table 12 – Results for TCN and LSTM for the different hyperparameter configurations when $h = 96$.

Method	RMSE_train	RMSE_test	Batch	k	d	b	Filters	l	Units
LSTM	3877.71	4066.99	32					1	32
LSTM	5172.98	4582.67	32					1	32
LSTM	3853.69	3672.89	32					1	64
LSTM	5456.27	4949.91	32					1	64
LSTM	4279.25	4264.62	32					2	32
LSTM	5292.85	4984.04	32					2	32
LSTM	3929.79	3833.94	32					2	64
LSTM	5738.95	5064.33	32					2	64
LSTM	5079.18	5141.33	32					3	32
LSTM	5861.93	5423.91	32					3	32
LSTM	4408.56	4369.82	32					3	64
LSTM	5883.98	5252.47	32					3	64
TCN	3573.98	3455.16	32	2	[1, 4, 12, 48]	1	32		
TCN	5611.99	4765.56	32	2	[1, 4, 12, 48]	1	32		
TCN	3631.37	3420.26	32	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	5358.19	4802.56	32	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	3778.72	3596.84	32	2	[1, 3, 6, 12, 24]	2	32		
TCN	5565.97	5410.53	32	2	[1, 3, 6, 12, 24]	2	32		
TCN	3662.52	3385.94	32	2	[1, 2, 6, 12, 24]	2	32		
TCN	5250.25	4548.51	32	2	[1, 2, 6, 12, 24]	2	32		
TCN	4084.47	4138.86	32	3	[1, 4, 16, 32]	1	32		
TCN	5435.08	4885.83	32	3	[1, 4, 16, 32]	1	32		
TCN	3727.74	3519.96	32	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	5327.74	5016.49	32	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	3667.05	3542.69	32	3	[1, 2, 4, 8, 16]	2	32		
TCN	5307.14	4942.50	32	3	[1, 2, 4, 8, 16]	2	32		
TCN	3763.31	3540.13	32	3	[1, 4, 16]	2	32		
TCN	5383.83	5006.96	32	3	[1, 4, 16]	2	32		
TCN	3635.52	3488.49	32	2	[1, 4, 12, 48]	1	64		
TCN	5692.10	4884.34	32	2	[1, 4, 12, 48]	1	64		
TCN	3592.46	3346.45	32	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	5758.89	5061.86	32	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	3624.47	3401.12	32	2	[1, 3, 6, 12, 24]	2	64		
TCN	5877.02	5484.75	32	2	[1, 3, 6, 12, 24]	2	64		
TCN	3619.55	3363.92	32	2	[1, 2, 6, 12, 24]	2	64		
TCN	5662.96	5432.22	32	2	[1, 2, 6, 12, 24]	2	64		
TCN	3963.92	3802.00	32	3	[1, 4, 16, 32]	1	64		
TCN	5778.44	5494.21	32	3	[1, 4, 16, 32]	1	64		
TCN	3982.00	3457.77	32	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	6025.36	5464.23	32	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	3735.19	3496.61	32	3	[1, 2, 4, 8, 16]	2	64		
TCN	5911.72	5257.16	32	3	[1, 2, 4, 8, 16]	2	64		
TCN	3725.60	3402.52	32	3	[1, 4, 16]	2	64		
TCN	5608.26	4894.99	32	3	[1, 4, 16]	2	64		
LSTM	4161.84	3963.22	128					1	32
LSTM	5345.83	4408.46	128					1	32
LSTM	3985.70	3875.17	128					1	64
LSTM	5273.64	4582.88	128					1	64
LSTM	4422.43	4252.99	128					2	32
LSTM	5256.39	4609.41	128					2	32
LSTM	4240.04	3877.81	128					2	64
LSTM	5384.12	4477.65	128					2	64
LSTM	4522.80	4336.65	128					3	32
LSTM	5252.03	4518.08	128					3	32
LSTM	4376.59	4316.72	128					3	64
LSTM	5351.07	4786.32	128					3	64
TCN	3679.93	3432.31	128	2	[1, 4, 12, 48]	1	32		
TCN	5314.93	4497.43	128	2	[1, 4, 12, 48]	1	32		
TCN	3637.41	3402.64	128	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	5438.66	4622.66	128	2	[1, 2, 4, 8, 12, 24, 48]	1	32		
TCN	3596.91	3333.10	128	2	[1, 3, 6, 12, 24]	2	32		
TCN	5421.31	4703.15	128	2	[1, 3, 6, 12, 24]	2	32		
TCN	3769.74	3483.55	128	2	[1, 2, 6, 12, 24]	2	32		
TCN	5504.00	4819.88	128	2	[1, 2, 6, 12, 24]	2	32		
TCN	3781.45	3500.03	128	3	[1, 4, 16, 32]	1	32		
TCN	5424.62	4519.38	128	3	[1, 4, 16, 32]	1	32		
TCN	3756.39	3523.69	128	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	5238.95	4718.58	128	3	[1, 2, 4, 8, 16, 32]	1	32		
TCN	3660.46	3436.77	128	3	[1, 2, 4, 8, 16]	2	32		
TCN	5302.32	4677.51	128	3	[1, 2, 4, 8, 16]	2	32		
TCN	3598.45	3351.58	128	3	[1, 4, 16]	2	32		
TCN	5426.24	4836.57	128	3	[1, 4, 16]	2	32		
TCN	3588.20	3379.56	128	2	[1, 4, 12, 48]	1	64		
TCN	5379.22	4635.63	128	2	[1, 4, 12, 48]	1	64		
TCN	3654.18	3423.69	128	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	5271.22	4536.82	128	2	[1, 2, 4, 8, 12, 24, 48]	1	64		
TCN	3672.72	3403.19	128	2	[1, 3, 6, 12, 24]	2	64		
TCN	5470.24	4829.98	128	2	[1, 3, 6, 12, 24]	2	64		
TCN	3709.85	3509.94	128	2	[1, 2, 6, 12, 24]	2	64		
TCN	5393.45	4676.39	128	2	[1, 2, 6, 12, 24]	2	64		
TCN	3710.02	3492.49	128	3	[1, 4, 16, 32]	1	64		
TCN	5226.96	4506.82	128	3	[1, 4, 16, 32]	1	64		
TCN	3654.12	3519.65	128	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	5235.32	4590.65	128	3	[1, 2, 4, 8, 16, 32]	1	64		
TCN	3773.80	3515.91	128	3	[1, 2, 4, 8, 16]	2	64		
TCN	5379.21	4806.03	128	3	[1, 2, 4, 8, 16]	2	64		
TCN	3695.33	3461.47	128	3	[1, 4, 16]	2	64		
TCN	5317.36	4630.92	128	3	[1, 4, 16]	2	64		

APPENDIX B – LOGISTIC REGRESSION LOSS CALCULATION

LR was used as Since in a logistic regression the loss function J is cross-entropy, it can be expressed as:

$$J = -\log p(y|x) = -[y \log \hat{y} + (1-y) \log(1-\hat{y})], \tag{B.1}$$

where $\hat{y} = \sigma(w.x + b)$. Replacing $\hat{y}$ in B.1:

$$J = -[y \log \sigma(w.x + b) + (1-y) \log(1-\sigma(w.x + b))] \tag{B.2}$$

The gradient of the loss function in relation to x can be expressed as:

$$\nabla_x J = \frac{\partial J(x)}{\partial x} \tag{B.3}$$

Adopting the natural logarithm as the log function, using the chain rule and considering that the first derivative of ln(z) is $\frac{1}{z}$,

$$\nabla_x J = -\left[y.\frac{1}{\sigma(w.x+b)}.\frac{\partial \sigma(w.x+b)}{\partial x} + (1-y).\frac{1}{1-\sigma(w.x+b)}\frac{\partial(1-\sigma(w.x+b))}{\partial x} \right] \tag{B.4}$$

Rearranging the terms, we obtain:

$$\nabla_x J = -\left[\frac{y}{\sigma(w.x+b)} - \frac{1-y}{1-\sigma(w.x+b)} \right] \frac{\partial(\sigma(w.x+b))}{\partial x} \tag{B.5}$$

$$\nabla_x J = -\left[\frac{y[1-\sigma(w.x+b)] - (1-y)[\sigma(w.x+b)]}{\sigma(w.x+b)[1-\sigma(w.x+b)]} \right] \frac{\partial(\sigma(w.x+b))}{\partial x} \tag{B.6}$$

$$\nabla_x J = -\left[\frac{y - y\sigma(w.x+b) - \sigma(w.x+b) + y\sigma(w.x+b)}{\sigma(w.x+b)[1-\sigma(w.x+b)]} \right] \frac{\partial(\sigma(w.x+b))}{\partial x} \tag{B.7}$$

The first derivative of the sigmoid function is given by:

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1-\sigma(z)) \tag{B.8}$$

Replacing Equation B.8 in Equation B.7, and rearranging the terms:

$$\nabla_x J = -\left[\frac{y - \sigma(w.x + b)}{\sigma(w.x + b)[1 - \sigma(w.x + b)]}\right] \sigma(w.x + b)[1 - \sigma(w.x + b)].w \tag{B.9}$$

$$\nabla_x J = [\sigma(w.x + b) - y]w \tag{B.10}$$

This last expression shows how the gradient of the logistic regression's loss function can be computed.

WORKS PUBLISHED BY THE AUTHOR

B.1 Papers Originated from Thesis

1. Everton J. Santana, Ricardo P. Silva, Bruno B. Zarpelão, and Sylvio Barbon Jr. Photovoltaic Generation Forecast: Model Training and Adversarial Attack Aspects. In *Proceedings of the 9th Brazilian Conference on Intelligent Systems (BRACIS)*, pp. 634-649, 2020.

B.2 Additional Works During Master's

B.2.1 Journals

1. Everton J. Santana, Felipe R. dos Santos; Saulo M. Mastelini; Fábio L. Melquiades; Sylvio Barbon Jr. Improved Prediction of Soil Properties with Multi-target Stacked Generalisation on EDXRF Spectra. *Chemometrics and Intelligent Laboratory Systems*, 209:104231, 2021.

2. Jéssica F. Lopes, Everton J. Santana, Victor G. T. da Costa, Bruno B. Zarpelão, and Sylvio Barbon Jr. Evaluating the Four-way Performance Trade-off for Data Stream Classification in Edge Computing. *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 1013-1025, 2020.

3. Saulo M. Mastelini, Everton J. Santana, Ricardo Cerri, and Sylvio Barbon Jr. DSTARS: A Multi-target Deep Structure for Tracking Asynchronous Regressor Stacking. *Applied Soft Computing*, 91:106215, 2020.

4. Everton J. Santana, João A. P. R. da Silva, Saulo M. Mastelini, and Sylvio Barbon Jr. Evaluation of Multi-target Regression to Support Decision on Stock Portfolio Investment. *iSys-Brazilian Journal of Information Systems*, 12(1):05–27, 2019.

5. Saulo M. Mastelini, Victor G. T. da Costa, Everton J. Santana, Felipe K. Nakano, Rodrigo C. Guido, Ricardo Cerri, and Sylvio Barbon Jr. Multi-output Tree Chaining: An Interpretative Modelling and Lightweight Multi-target Approach. *Journal of Signal Processing Systems*, 91(2):191–215, 2019.

6. Gabriel J. Aguiar, Everton J. Santana, André C. P. F. L. de Carvalho, Sylvio Barbon Jr. Using Meta-Learning for Multi-target Regression. *Information Systems (Under Review)*.

7. Gabriel J. Aguiar, Everton J. Santana, Rodrigo G. Capobianco; Mario L. Proença, Sylvio Barbon Jr. Multiple Voice Disorders in One Individual: investigating bio-inspired features, multi-label classification algorithms and base-learners. *Computers and Electrical Engineering (Under Review)*.

B.2.2 Book Chapter

1. Sylvio Barbon Jr, Everton J. Santana, Amanda T. Badaró, Nuria A. Borrás, and Douglas F Barbin. Advantages of Multi-target Modelling for Spectral Regression. In: A. K. Shukla (ed.). *Spectroscopic Techniques & Artificial Intelligence for Food and Beverage Analysis*. Springer Nature Singapore Pte Ltd., 2020.

B.2.3 Conferences

1. Gabriel J. Aguiar, Everton J. Santana, Saulo M. Mastelini, Rafael G. Mantovani, and Sylvio Barbon Jr. Towards Meta-learning for Multi-target Regression Problems. In *Proceedings of the 8th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 377–382. IEEE, 2019.

2. Victor G. T. da Costa, Everton J. Santana, Jessica F. Lopes, and Sylvio Barbon Jr. Evaluating the Four-way Performance Trade-off for Stream Classification. In Rodrigo Miani, Lasaro Camargos, Bruno Zarpelão, Erika Rosas, and Rafael Pasquini (editors),*14th International Conference on Green, Pervasive and Cloud Computing - Lecture Notes in Computer Science*, volume 11484, pages 3–17. Springer, 2019 - Honarable Mention Award.