

École polytechnique de Louvain

Deep Neural Networks for Spatio-Temporal Meteorological Prediction

Author: **Adrien BUCHE**

Supervisors: **Michel VERLEYSEN, Pierrick BRUNEAU**

Readers: , **John LEE, Benoit OTJACQUES**

Academic year 2018–2019

Master [120] in Data Sciences Engineering

Université Catholique de Louvain

Abstract

Ecole Polytechnique de Louvain
Master in Data Sciences Engineering

Deep Neural Networks for Spatio-Temporal Meteorological Prediction

by Adrien Buche

For centuries, **Meteorological prediction** has been a challenging task every civilization faced. With the development of recent technologies, it became easier to obtain accurate weather forecast. The objective of this thesis is to apply **Deep Neural Networks** algorithms on meteorological data to precisely predict the solar irradiance throughout the day. This is a crucial step in the optimization of renewable energy production and management. With the dawn of high performance computational hard-wares and the availability of large datasets, Recurrent Neural Networks and Convolutional Neural Networks have been built to predict the solar irradiance ratio one hour into the future. Thanks to a dataset composed by a dozen of variables recorded during five years, we built models to predict the desired value on five different stations on the *La Reunion* island. Compared to current state of the art algorithms, our models were able to predict accurate and truthful sequences, decreasing the error by 25% in the best case, with **mae**'s and **rmse**'s metrics reaching values as low as 0.17 and 0.19, respectively. Additionally, we experimented new types of algorithms, combining the data from multiple stations to build spatio-temporal models or combining the recurrent and the convolutional layers into one models. They gave promising results but necessitate a fine tuning of the parameters! During this work, it showed Deep Learning algorithms could improve the current solutions for meteorological predictions and that new data-driven solutions, for complex problems, will be discovered in the future as data become more and more available.

Acknowledgements

First of all, I would like to thank my academic supervisor Michel Verleysen. Throughout this year, he gave me the motivation to tackle this important work. His advice and comments were really helpful during the writing of the thesis.

During this research work, I spent four months inside the *Luxembourg Institute of Science and Technology*. I want to thank Pierrick Bruneau for his dedicated attention during my internship. He provided the technical support I needed and he helped me formulate ideas and experiments following the correct procedure. I am grateful to all the persons I met at *List* including Marc-Henri Comans, Nicolas Médoc, Benoit Otjacques and all my others colleagues from the *ERIN* department.

A special thanks to Mathieu Delsaut from University of *La Réunion* for the help he offered to access additional data to conduct my work.

I would like to sincerely thanks my parents, my sisters Lisa and Madeleine and my girlfriend Fannie for their contributions to this thesis and during the five years I spent at U.C.L.. They brought me joy, comfort and moral support during 5 years so that I can fully dedicate myself to my studies and to carry out the Master thesis. My grand-parents, aunts, uncles, cousins and close friends were a great support during my studies, they helped me whenever I needed them.

During my master degree, I met two really important persons. Anat and Akvile, I am grateful to be part of your close circle of friends ! These two years would not have been the same without you! I want to thank you for all the help you gave me and all the things I could learn throughout your experiences that you willingly shared with me! Thank you and your families.

Summary

Abstract	i
Acknowledgements	ii
1 Introduction	1
1.1 Motivations	2
1.2 Historical Overview and State of the Art	3
1.3 Thesis Structure and Contributions	5
2 Theory	6
2.1 Time Series	6
2.2 Models	10
2.2.1 Neural Networks	10
2.2.2 Recurrent Neural Networks	14
2.2.3 Convolutional Neural Networks	18
2.2.4 Convolutional Recurrent Neural Networks	20
3 Data Analysis	21
3.1 Data Exploration	21
3.2 Pre-processing	28
3.3 Additional Variables	30
3.4 Dataset construction	31
3.5 Discussions	32
3.5.1 Target Distribution	32
3.5.2 Class Imbalance	34
3.5.3 Past and future knowledge	35
4 One-hour Prediction Task	36
4.1 Methods	36
4.1.1 Models	36
4.1.2 Training and test phases	37
4.1.3 Performance metrics	38
4.1.4 Parallel and Cross site	38
4.2 Results	39
4.2.1 Recurrent Neural Networks	39
4.2.2 Convolutional Neural Networks	42
4.3 Improved methods	45
4.3.1 Spatio-Temporal Combinations	45
4.3.2 CRNN	51

4.3.3 Discussions	51
5 Conclusion	56
Appendices	59
A Activation Functions	60
A.1 Sigmoid Function	60
A.2 Hyperbolic Tangent Function	60
A.3 Rectified Linear Unit Function	61
B Models Structure	62
B.1 Recurrent Neural Networks	63
B.2 Convolutional Neural Networks	64

List of Figures

2.1	Monthly <i>Wolf</i> number (# of sunspots) variation from 1749-1983	7
2.2	Tesla Model Sales from 2015	7
2.3	Anscombe's quartet : the 4 dataset share all the same mean for x's, 9, and y's, = 11, as well as they all have the same variance for x's ,(7.5), and y's ,4.125, however, as one can see, they are all intrinsically very different. [1]	9
2.4	Artificial Neural Network Structure [2]	10
2.5	Comparison of two types of neuron	12
2.6	Recurrent Neural Network Representations	14
2.7	LSTM Representation	16
2.8	Recurrent Equations following the order in which they are used within the LSTM cell	17
2.9	Forward propagation of the input values over several time steps. Each color of a node refers to the information passed along the different states over time. Small bits are kept, others are forgotten, as said before.	17
2.10	Back-propagation of the error within the entire network of neurons. The red color panel indicates which neurons are affected by the retro-propagation.	18
2.11	Convolutional Neural Network Structure	19
3.1	La Reunion Island	21
3.2	Sunlight Characteristics	22
3.3	Bar chart of the humidity ratio for each station, represented by its dedicated color. The $x - axis$ represents the percentage of humidity in the air while the $y - axis$ is the number of observation with the respective value.	24
3.4	Keeping the same color legend for the stations, this bar chart describes the number of occurrences, within each station, of the direction of the wind.	25
3.5	Scale for Correlation	26
3.6	Correlation Tables	26
3.7	Auto-correlation	28
3.8	Dataset Transformation	32
3.9	Scale of Kb ratio	33
3.10	Target Distribution by Station	34
3.11	Imbalance of the target	35
4.1	Dataset Separation	37
4.2	Spatio-temporal Model's Structure	45

4.3	Small Multiples MAE Results Representation for the Spatio-Recurrent Models on <i>Possession</i> . The x-axis of each sub-figure characterizes the set ID present in Table 4.7.	47
4.4	Small Multiples MAE Results Representation for the Spatio-Convolutional Models on <i>Saint-Leu</i> . The x-axis of each sub-figure characterizes the set ID present in Table 4.8.	50
4.5	Small Multiples MAE Results Representation for the Spatio-Convolutional Models on <i>Possession</i> . The x-axis of each sub-figure characterizes the set ID present in Table 4.8.	50

List of Tables

3.1	A table outlining the 5 stations geographical characteristics.	21
3.2	Descriptive Statistics for the five stations.	23
3.3	Correlation between sites	27
4.1	Top 3 RNN Models	41
4.2	Best MAE for RNN models	42
4.3	Best RMSE for RNN models	42
4.4	Top 3 CNN Models	43
4.5	Best MAE Models CNN	44
4.6	Best RMSE Models CNN	44
4.7	Sets of RNN parameters	46
4.8	Sets of CNN parameters	48
4.9	Best results for CRNN models	51
4.10	Final Results & Comparison	52
B.1	Structure of the RNN models	63
B.2	Structure of the CNN models : Part 1	64
B.3	Structure of the CNN models : Part 2	65
B.4	Structure of the CNN models : Part 3	66

Chapter 1

Introduction

Meteorology has always been a human preoccupation. Activities, like farming or hunting, strongly depend on weather conditions, which caused humans to observe meteorological phenomena to forecast future events. Many societies considered weather conditions as an act of gods and looked after priests to predict or influence the weather. Though the origin of Meteorology forecasting is impossible to accurately define. Meteorology itself separates into two distinctive branches, a branch of **science** and a branch of **knowledge**. The first signs of Meteorology predictions, as a branch of knowledge, appeared with the first human civilizations, dating back millennia. . Greeks (7th century B.C.) were the first to consider meteorology forecasting as a branch of sciences. *Thales of Miletus* associated weather with movements of stars and planets [3]. Later on around 340 B.C., *Aristote*, considered as the first meteorologist (i.e. a weather forecaster), introduced basic elements such as water, earth, wind and fire to be the key elements of weather regulations [4].

Due to the Roman dominance over the Greek civilizations, for centuries, Greek meteorologists who were the main "scientists" to focus on this subject, could not carry out their works. Hence, for 2000 years after *Aristote*'s findings, meteorologist practices resumed to be series of factual observations, throughout surveillance of cloud or stars movements, speculation of thunder and lightning or relations between the fundamental elements but no one added anything significant to the meteorologist field ! In 1593 only [3], the first **thermometer** was invented by *Galileo*. At first, instruments like rain gauges, anemometers, barometers appeared and then, with the enhancement of general technology, more sophisticated tools were designed. During the 20th Century, thanks to the evolution of machines like radars, satellites, weather stations and computers, the accuracy and efficiency of weather forecast was greatly improved and thereafter, it kept getting better and better.

In contrast to the evolution of Meteorology, the evolution of Machine Learning was much faster. The dawn of Machine Learning, enabled by the arising capabilities of fast computing, gave birth to new kind of algorithms and methods to approach problems from a different perspective and tackle big data related challenges. Since the 50's, supervised learning, including prediction and classification, or unsupervised learning, like data-mining or information retrieval, allowed researchers and scientists to discover new types of solutions to problems that could not be solved before. Machine learning algorithms also improved existing solutions and allowed better optimization of many

processes. In the financial sector, *American Express* uses Machine Learning algorithms to detect fraud among transactions in real-time saving millions of dollars. The car company *Volvo* uses artificial intelligence to predict the failure of car components, enabling early car revisions [5]. These 2 examples, among many others, show the great potential of Machine Learning for the society.

Witnessing the recent improvements in commodity and High Performance Computer hard-wares, e.g. libraries to exploit those like *CUDA*¹ and breakthroughs in Machine Learning algorithms bound to these libraries, joining these processing capabilities with the meteorological data, collected throughout multiple channels, demonstrates an important potential for this field of sciences driving our everyday life.

1.1 Motivations

Since the 16th Century, philosophers used tools previously mentioned to systematically record weather variations. All these observations served a common purpose : to forecast the future weather conditions. In accordance with the development of new technologies, weather predictions became more and more precise, but at the price of costly numerical simulations.

In cooperation with the *Luxembourg Institute of Science and Technology* (LIST), we wanted to investigate the possibilities using terrestrial data, e.g. data recorded thanks to automated surface-observing systems installed on Earth, to predict solar irradiance taking advantages of Deep Neural Networks' capabilities such as their pattern-matching techniques to analyze vast quantities of supervised data. *LIST*'s Environmental Research and Innovation (ERIN) research department, and more specifically, the Environmental Informatics unit is specialized in data driven research, development and innovation within the field of environment informatics to provide solutions for a more sustainable future [6]. Accordingly to this guideline, the main objective of this thesis is to predict the solar irradiance ratio with an horizon of one hour, given the last 60 minutes of observations.

The main purpose of predicting solar irradiance accurately is to optimize the management of photo-voltaic power plants which is a hot topic in the renewable energy production industry. On one hand, satellite imagery offers useful information for weather prediction. Nonetheless, it requires a comprehensive analysis, either done by humans or algorithms, of the generated images to obtain complete results. On the other hand, Numerical Weather Simulations, based on physics models, provide most of the prediction power in the current industry. As an alternative to these time-consuming procedures, our intentions behind applying deep learning models on meteorological data, are that we want to implement algorithms capable of efficiently and quickly replicating human brain capabilities like recall and recognition, i.e capacity to detect and remember important events from the past, given the appropriate data. Furthermore, depending on the utilization of the predicted variable, the interval between two predictions could be relatively small, from a minute or larger to few hours or days.

¹<https://developer.nvidia.com/cuda-zone>

Therefore, models need to be easily adaptable and algorithms such as time-series deep neural networks bring hopeful perspectives to fit these requirements.

The increasing demand of solar energy around the world is one of the reasons so many researches have been conducted over the years. By the end of the 21st Century, solar energy requirements and application are expected to triple [7]. An efficient and fast forecasting method that can be applied anywhere in the world is necessary to help the renewable energy sector to correctly face the challenges of tomorrow. Whether it is supply management or electricity grid operation controls, the forecast of solar energy will partially determine the economic prosperity and quality of life of the future generations. The adaptability of the spatio-temporal models, multiple local weather stations working together studied throughout this work could be a step towards a more precise, hence cleaner forecast of the solar energy.

1.2 Historical Overview and State of the Art

Consequently to the rapid growth of renewable energies, the production and consumption of solar power as a source of electricity have surged over the recent years. For this reason, a large interest has risen among the scientific community in forecasting solar radiation over short time horizons. Although this topic brought a lot of philosophers and scientists to put their minds to it, we will review the works of some of them over the centuries.

Around 1513, *Nicolas Copernic* (1473 - 1543) suggested the theory that the sun is the center of the system, around which the solar system's planets revolve. This observation was one of the first reflection based on spatial physics laws. With other modernized laws, defining how the atmosphere behaves and interacts with environmental elements, the Numerical Weather Prediction (NWP) uses mathematical models based on physics and chemistry, to predict the oncoming weather conditions based on the present events using super-computers to run simulations. They were firstly initiated by *Vilhelm Bjerknes* [8]. Later on, researches related to solar forecasting problems, were based on physics laws and modern NWP to tackle this more and more important challenge (see Perez and al. [9], Molineaux and al. [10], for examples).

Additionally, due to NWP's complexity and cost, others scientists turned to statistical approaches "Model Output Statistics" [11], that are a multi linear regression technique, or generated "Markov transition matrices" [12], to predict future sequence of solar activities. More specially, in [13], Goh and Tan (1977) explored statistical models to predict solar variations. Through stochastic modelling theory, they were the first ones to pioneer short-term models compared to models like [14] or [15] which were only designed to predict long-term forecasting and were lacking assumptions about the correlated nature of the solar data. With the help of satellite data, Hammer and al. [16] used a statistical approach to predict clouds' behaviors (motion, formation, dissolution) and thus, the irradiance ratio. Auto-regressive models, used to represent time-varying processes in nature or economics, have also been applied to the prediction of solar forecast (for instance, in Safi and al. [17] or Zhang and al. [18]). Compared to physics-based models, these techniques are data driven. They use the data from past

events and try to make inference from it. These approaches have left the door open to more modern and advanced data driven methods using the features present in the data to the full potential, e.g. Deep Learning Algorithms.

More recently, Machine Learning methods rose interests in many science fields, thanks to a surge of data availability and fast computation capabilities. Gladly, weather predictions models weren't left aside. Models, among many others, like Regression Trees [19], Support Vector Machines (SVM) [20] or Neural Networks ([21], [22], [23]) profit from the learning capabilities of machine learning models to determine if relations, linear or non linear, exist between all the variables. These solutions for solar irradiance prediction have demonstrated better performance than statistical or auto-regressive models [24], achieving results as good as rmse's errors² between 10% to 30%, which give excellent overview of their capabilities.

Finally, (Deep) Neural Network researches have emerged to tackle the prediction of the solar irradiance. Thanks to the temporal scaling of data, Recurrent Neural Networks have proven to be more accurate and more consistent than basic Neural Networks, like the multi-layer perception, for short and long term prediction. For example, Barbounis and al. [25] have used Deep Neural Networks to predict long-term wind speed and solar power at an hourly rate. Alzahrani and al. [26] also compared the performance of different models, namely, support vector regression, feed-forward neural networks and deep recurrent neural networks. They all came to the conclusion that deep learning neural networks can easily handle massive amount of historical data and that the recurrent feedback over time is a great model's feature when dealing with weather prediction tasks.

Another kind of Deep Neural Network is the Convolutional Neural Network. As a result of their filtering and pattern recognition capabilities, Convolutional Neural Networks have demonstrated good levels of efficiency in the treatment of satellite imagery. In [27], Y Liu and al. applied Convolutional Neural Networks on climate datasets to detect meteorological extreme events variations. Still based on 2D images datasets, Klein and al. [28] presented a generalized version of a convolution layer to analyze weather conditions at an interval of 10 minutes. Nevertheless, no real experiments have been conducted on 1D meteorological data sequence with the former type of model to predict the solar irradiance ratio. However, it showed impressive performances on other 1D related problems like speech recognition [29] and natural language processing. These milestones suggest that Convolutional Neural Networks should also be given a try. They will be used in the studies of data driven solar predictions, comparing them to Recurrent Neural Networks.

With so many researchers implementing solutions on this topic, the need for a precise and persistent forecast of the solar irradiance is an important matter for the future of our society. The renewable energy industry is yet to evolve to a more efficient and profitable market, bringing a lot of attention from the scientific community. Deep Learning models offer many advantages over classical Machine Learning techniques such as the capability to deal with large dataset and to provide the associated computing power. Mastering higher levels of data representation, they can solve more

²Explanations about this metrics' characteristics and purposes will follow in a later section.

complex problems using special kinds of datasets. This is exactly why these types of models will be the main focus of the present work.

1.3 Thesis Structure and Contributions

Starting with Chapter 2, theoretical aspects focusing on some details and key characteristics of Neural Networks are introduced. This is an effort to set all readers on the same level of knowledge. Therefore, we will go through the basic components enabling a Neural Network to learn from data. Further explanations on the Deep Neural Networks, the recurrent and the convolutional neural networks, will be brought to the users to understand how they differ from each others and how they are improved versions of the classical neural network.

In Chapter 3, an in-depth analysis of the data used during this work has been conducted. In the Machine Learning field, it is really important to consider the variables from different angles, so that their utilization to build the model, is optimized, well understood and not mistaken. From data exploration to pre-processing and discussions about the target variable, this chapter contributes to the discovery of every possible feature of the dataset such that suitable models' structures are considered during the research.

Then, Chapter 4 tackles the main part of the present research task. Supervised Machine Learning is all about prediction or classification of future events, provided some past observations and their outcomes. To do so, initially based on a **sequence** of past 60 minutes observations, our first goal is to predict a specific variable, the solar irradiance ratio, one hour after the most recent observation recorded. With the help of sensors situated at different locations on the *La Reunion* island, variables like the temperature, the atmospheric pressure or the humidity ratio were recorded with a time interval of one minute during five years. Several strategies, mentioned in Chapter 2, are investigated to exploit long-short term recurrent events or pattern recognition of temporal events to efficiently predict the targeted value. The second objective is to explore the possibilities of spatial combinations between the weather stations to understand how they interacts between each other and if the results are better then the single station models.

Finally, Chapter 5 summarizes the main results of the research and introduces potential improvements to the current models or additional leads to capitalize on the present work.

Chapter 2

Theory

In this chapter, we will mainly introduce the theoretical knowledge one needs to correctly understand the answers of this thesis' research question, i.e. how will we use the Deep Neural Network framework to predict the solar irradiance ratio with an horizon of one hour, based on the last hour of available data. We first present the concept of *time-series*, that is the specific type of data employed through out this project. Then, theoretical notions about the models, used to process the data, are described. Below, we suggest generic explanations about *Neural Networks* models. Then, speaking about two specific categories of these models, the *Recurrent Neural Networks* [RNN] and the *Convolutional Neural Networks* [CNN], details and technicalities about each type will be presented to the readers.

2.1 Time Series

Introduction An intuitive definition is : A **time-series** is a sequence of observations x_t , each one being recorded at a specific time t [30]. A time-series can either be *discrete*, the observations are constructed at fixed time intervals, or *continuous*, the observations are recorded continuously over a time interval. Another important characteristic of time-series is that the observations can be **deterministic** or **stochastic**. On one hand, a deterministic time-series indicates the sequence of observations is regulated, by a physic law or a known phenomenon, and that the future predictions could be determined in advance. On the other hand, the future observations of a stochastic time-series can only be partly determined based on the knowledge of the past values. This main assumption clarifies that, in the past, some scientists experienced physics-based models to study time-based meteorological processes and why others, tried models based on the previous data. This inner characteristics will probably have a huge effect during the performance of the prediction task.

Once the observations have been recorded during a sufficiently long period of time, the main task is to analyze the time-series of each variable. Before talking about the **analysis** task, let's take a look at two simple examples of time-series.

Examples Below, two examples of discrete time series are shown : One describes the monthly number of sunspots [31], the *Wolf* number, and the other defines the monthly sales, in the US, of different Tesla models [32].

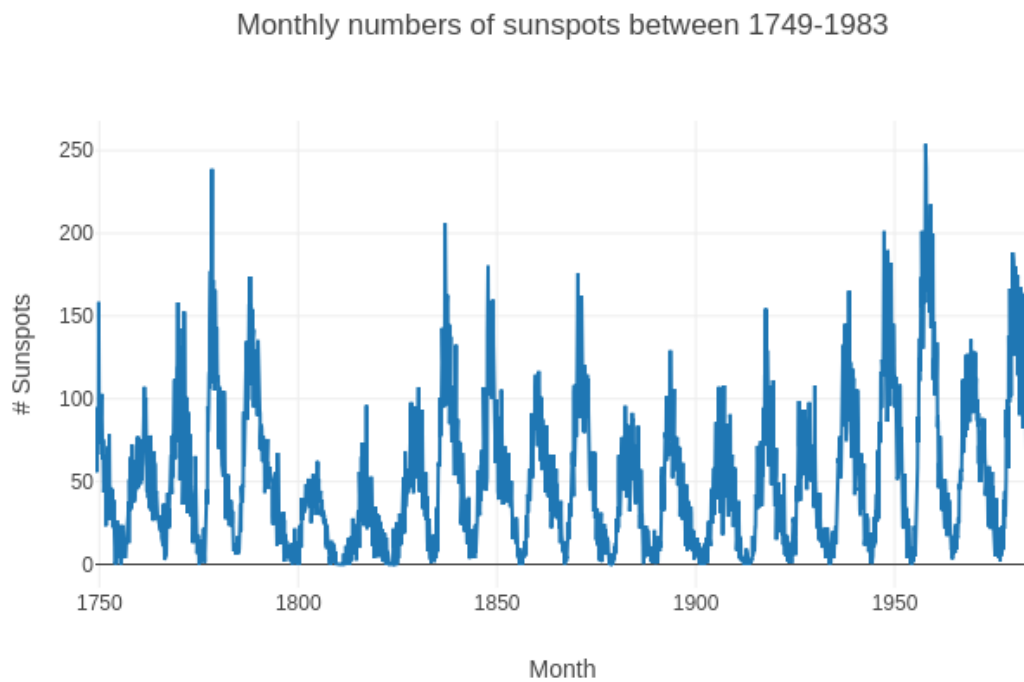


Figure 2.1: Monthly *Wolf* number (# of sunspots) variation from 1749-1983

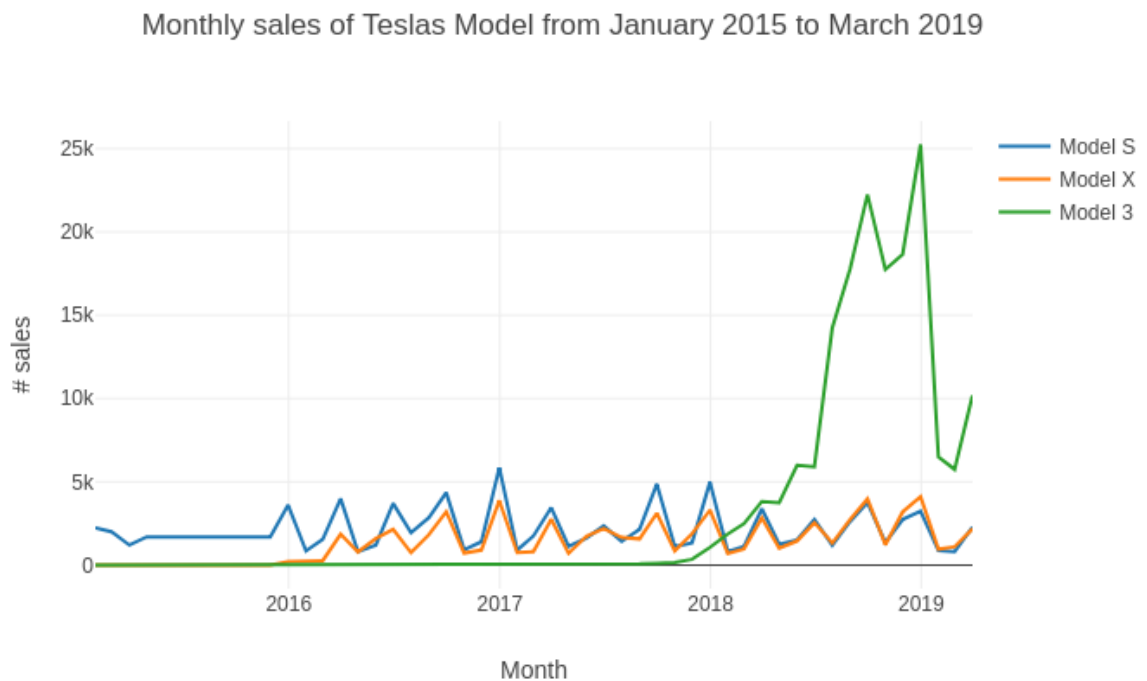


Figure 2.2: Tesla Model Sales from 2015

Looking at these graphs, one can understand the interest for time series. On Figure 2.1, we can clearly identify a recurrence and an increase of the number of spots over the years. On Figure 2.2, one can quickly analyze the differences between the car model's sales and also detect when an event happens (like the surge in the sales of the *Model 3*). Such graphics allow the users to control and monitor the evolution of designated targets (like the cars' sales) or to detect anomalies over time. Truly continuous time-series are more rarely encountered, however, in the medical field, electroencephalograms (EEG) are an example of continuous monitoring of the brain activity. Another example, in seismic surveys, controlling the state of volcanoes or tectonic plates is a way to predict the forthcoming and possibly destructive events of such activities.

Objectives As stated above, time series are reported so that one can analyze them and makes inferences. However, it exists four primary objectives in analyzing a time series :

- Description** consists of obtaining simple descriptive measures (mean, variance, correlation) of the variables, identifying the trends, variations or outliers, etc, ... All these intermediate tasks, being executed with simple models, give a general overview of the data before it is processed.
- Explanation** is based on the previous descriptions. It can either look for the consequences and/or reasons why such observations were made. Or it can be used to interpret the variables' behaviours and effects on others, like the reasons behind the behaviour of a variable inducing, or caused by, another variable's behaviour. This analysis leads to a deeper and better understanding of the general mechanism dictated by the variables' values.
- Prediction** will try to determine the values of future observations based on the past data. Many models and methods exist to forecast the forthcoming sequences, depending on the type of the series (deterministic or stochastic) and on the domain of application.
- Control** is closely related to the previous objective, the *prediction*. Indeed, controlling procedures (of tools or mechanism) can greatly be optimized knowing what will occur in the future. The usage of these tools and machines can also be optimized (increasing the lifespan and the way to use it, for example). Natural phenomena can also be supervised thanks to these controlling measures. Nevertheless, none of the above actions is possible if the predictions are erroneous. The need of an accurate and relevant forecast is crucial.

These four main objectives must logically be executed in sequence. It wouldn't make any sense to predict future observations without knowing how the past observations behaved.

Analysis Many different descriptive techniques can be executed during the description step in order to better understand the data. Some of them, used later on during the third chapter, are explained below.

The first easy and simple step is to compute the statistical metrics, such as the mean, variance, median and quantiles. This allows the user to understand the general behaviours and ranges of the data. These metrics can, readily, indicate whether or not some variable's ranges present outliers or gaps within the time series. In such cases, methods like outliers removals (or transformations) or (non-) linear interpolation can be applied on the dataset to fix these "imperfections". However, while gaps in sequences are really troublesome, outliers express some heterogeneity in the dataset that can benefit some prediction models. Likewise, it is also important to visualize the data with different kinds of charts. It has been shown in [33], that datasets with similar simple descriptive statistics, appear to be really different once plotted [1], this could be the case for time series. Obviously, the data of the fourth quartet (X_4) do not represent a time series sequence.

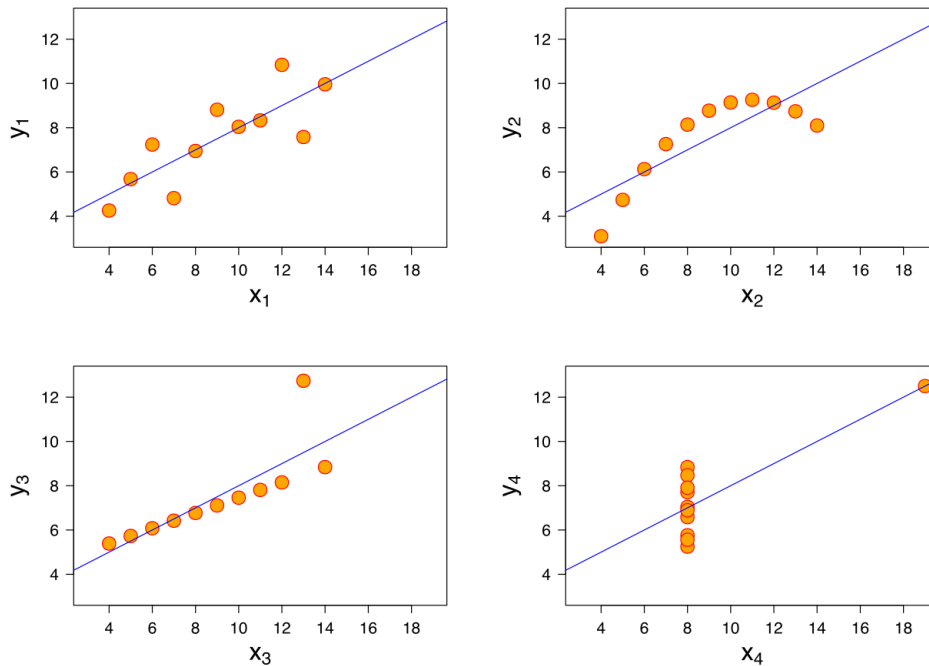


Figure 2.3: Anscombe's quartet : the 4 dataset share all the same mean for x's, 9, and y's, = 11, as well as they all have the same variance for x's, (7.5), and y's, 4.125, however, as one can see, they are all intrinsically very different. [1]

Other metrics such *correlation*, that consists of an analysis of a linear relation between two different variables, or *auto-correlation*, that is an analysis between observations within a variable time-line, can outline trends or seasonal recurrence. Indeed, the purpose of this metric will compute the correlation between a variable's record and a delayed copy of itself. Tuning the delay allows the recognition of different types of time series: random, short-term, alternating, non-stationary (trends) or seasonal fluctuations time series. All these tests and methods will be use on the data to gain information about it.

2.2 Models

The next section describes the theoretical models used to deal with the time series prediction linked to the research question. Many models already exist to tackle this important problem. This work particularly focuses on one category, the **Deep Neural Network** models. This kind of models differs from the other ones (Trees, Support Vector Machines or Multi-Layer Perceptron) for the following reasons. First, it uses a structure of multiple layers built with units, allowing the extraction and the transformation of the information from the input data. It can be used in both *Supervised Learning*, regression or classification task, and *Unsupervised Learning*, patterns detection. Finally, thanks to the multiple levels of layers and the specificities of their inner cell units, the Deep Learning models can detect and represent different levels of abstraction within the data features. This type of models is specially interesting because it replicates some human capabilities that no other model is able to implement. It is adequate to deal with specific types of input like images or sequences.

First, general concepts of (Deep) Neural Networks will be introduced such as : what is a Neural Network? How does it work?, etc,... Then, we will continue with some explanations, focusing more on the two specific models of Deep Neural Networks, *Recurrent Neural Network* [RNN] and *Convolutional Neural Network* [CNN]. What differentiates them from the general Neural Network form ? What are their specificities ? etc,...

2.2.1 Neural Networks

This might be quite of a strange word for the uninitiated reader, more related to biology then to computer sciences. This term is indeed related to biology as the Neural Networks models, used in computer sciences, borrow many similarities from the human brain, like the neurons itself or the connections between them.

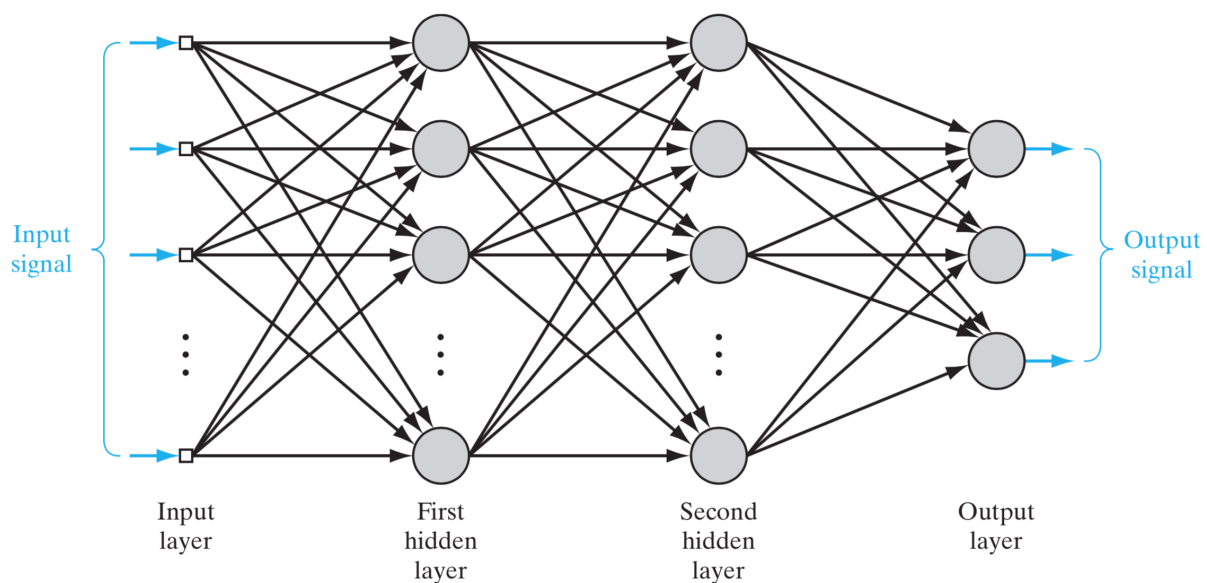


Figure 2.4: Artificial Neural Network Structure [2]

Structure A Neural Network is characterized by 3 types of layers : the input layer, the hidden layer(s) and the output layer. Each layer has the same overall structure : it is a stack of neurons, connected with all the neurons from the previous and next layers but never with the ones from the same layer (no inner connections). As a result of this general layout, such structures are called neural networks, mimicking our human brain. Below is a description of all three layers that one can observe on Figure 2.4.

Input : This layer is the "front door" of the model. The data feeding the model will be inputted through this layer. The format of input is always a vector of numbers. Consequently, images or matrices will require some pre-processing beforehand.

Hidden : This type of layer is truly where the model learns the representation of the data. Thanks to the layers of neurons, the model computes and adjusts the value of all the weights interconnecting the neurons together. The weights genuinely represent the knowledge of the model. As it is the case in real life, it exists many ways to learn and thus, many ways to build models.

Output : The outcome of the model will be produced at this Output layer. In case of a regression model, this layer contains only one neuron, an unique value is produced. In case of a classification, this layer contains as many values as the target observations include labels. Each one of the value represents the probability of the input being part of the respective label's class.

Brain Analogy As said before, the Neural Network models correspond to the Neural Network present in our brain. The different types of layers of the computer models can be interpreted as the different parts of the brain : picking up the input information thanks to the different lobes, treating information thanks to the interconnections between the lobes and then, producing a result.

To make that possible, our brain's primarily processing unit is called the neuron (see on Figure 2.5a). Paying attention to its structure, one can see *dendrites*, a *cell body*, an *axon* and its terminals, also called *synaptic terminals* [34]. Now, comparing the biological structure to the artificial one (on Figure 2.5b), one discerns the clear correspondence. The connections from the input signal to the summing function node are interpreted as the *dendrites*. Then, the computation steps are done in the cell body of the neuron (both biological and artificial), where in the artificial neuron, it is represented by the activation function bloc. Finally, once the result needs to be transmitted, it is passed along by the *axon* or another weighted edge in the artificial brain (where there are no differences between the input and output connections).

Finally, both brains maintain the connectivity between the processing cells thanks to millions and millions of connections. In [36] and [37], they state the human brain contains, on average, 100 billions of neurons and each neuron has approximately 7000 synaptic connections. The artificial neural network also mimics this property. Every neuron of a layer is fully connected to all neurons of the previous layer as well as to

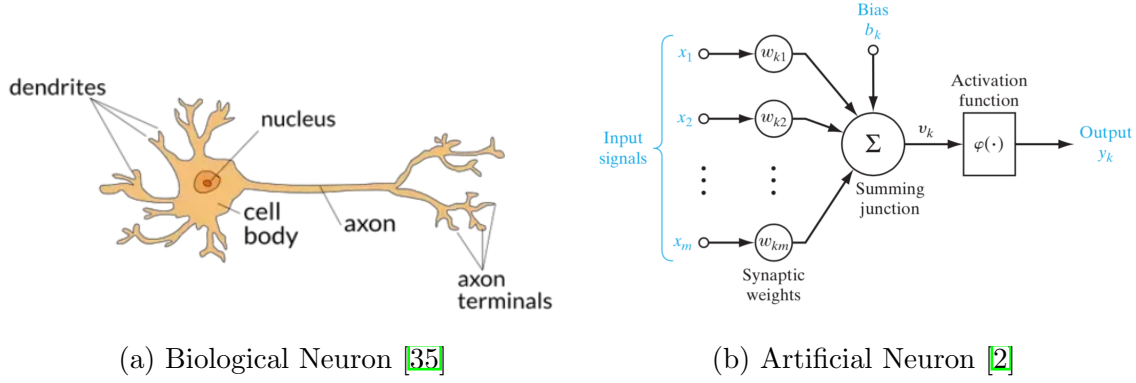


Figure 2.5: Comparison of two types of neuron

all neurons of the next layer (as you can notice on Figure 2.4). The purpose of this structure supposedly provides a better flow of the information among the network.

Activation Function After discussing the structure of the artificial network, let's summarize how the artificial cell works. All the input values x_j , weighted by their respective weights w_{kj} , are linearly combined and form a new value v_k (2.1), the linear combiner output. Then, the result of the summing function will pass through an **activation function**. This step regulates the amplitude of v_k and the outcome continues either to another hidden layer or to the outcome layer. Notice also that a bias b_k (positive or negative) can be added to v_k , affecting the net input of the activation function (2.2). Some examples of activation functions, ρ , are presented in Appendix A. It includes the *sigmoid*, the *hyperbolic tangent* or the *ReLU* functions.

$$v_k = \sum_{j=1}^m w_{kj} x_j \quad (2.1)$$

$$y_k = \rho(v_k + b_k) \quad (2.2)$$

Back-propagation Once an input has been given to the model, it produces an output. Afterwards, the model uses this predicted target value $o_j = y_j^{(L)}$, where L is the output layer, to compute the error signal $e_j = d_j - o_j$, with respect to the true target value d_j . Then, the weights are updated thanks to the generalized delta rule below:

$$w_{ji}^{(l)}(t+1) = w_{ji}^{(l)}(t) + \Delta w_{ji}^{(l)}(t) \quad (2.3)$$

$w_{ji}^{(l)}(t+1)$ is the synaptic correction of the weight $w_{ji}^{(l)}(t)$. The second term on the left, e.g. the gradient of the weight, can be computed using the chain rule of calculus, resulting in the equation below :

$$w_{ji}^{(l)}(t+1) = w_{ji}^{(l)}(t) + \eta \partial_j^{(l)} y_i^{(l-1)}(t) \quad (2.4)$$

In this formula, $y^{(l-1)}(t)$ is the input value of the neuron and η is the learning rate parameter of the algorithm. It defines how important are the adjustments of the

weights with respect to ∂ . Speaking about $\partial_j^{(l)}$, it represents the local gradient at the layer level. Depending on the layer level of the back-propagation, this gradient can be computed in two possible ways : Equation (2.5) refers to the calculus when the back-propagation is at its early stage, the output layer L . Equation (2.6) specifies how the local gradient is computed at a hidden level, $\forall l \neq L$. The indices $(t+1)$, (t) , $(t-1)$ indicate which iterated values of the weights are used to compute the updated weight and the (l) shows which layer the back-propagation is currently.

$$\partial_j^{(l)} = e_j \cdot \rho'(v_j) \quad \text{if } l = L \quad (2.5)$$

$$\partial_j^{(l)} = \rho'(v_j^{(l)}) \cdot \sum_k \partial_k^{(l+1)} w_{kj}^{(l+1)} \quad \forall l \neq L \quad (2.6)$$

In the formulas above, the back-propagation at the output layer is only affected by the unique value the neuron had just emitted. However, at an hidden level, the weight of each neuron is revised by all the weights of the previous layer (in the backward order) it is connected with. On figure 2.4, the bottom neuron of the first hidden layer will be impacted by the weights' values of the second hidden layer neurons it is connected with, which means $(l+1)$ layer. ρ' is the derivative of the activation function used at that specific layer, since each layer could have its own activation function.

An alternative formula of the back-propagation algorithm includes an additional term, the momentum α . In Equation (2.7), the momentum term is a constant, controlling furthermore how the adjustments behave. It is particularly useful when one wants to attenuate the oscillations of the gradient descent operated during the back-propagation. In further chapter, we will introduce several algorithms' implementations corresponding to the methods mentioned here.

$$w_{ji}^{(l)}(t+1) = w_{ji}^{(l)}(t) + \eta \partial_j^{(l)} y_i^{(l-1)}(t) + \alpha [\Delta w_{ji}^{(l)}(t-1)] \quad (2.7)$$

Finally, while this process of forward and backward propagation is repeated over many iterations, it allows the model's weights to learn the data representation and consequently, to potentially make better and more accurate predictions, e.g. *the model is learning!* However, in doing so, one must monitor how the training is performing over the iterations. With a low number of epochs, the model could **under-fit** the data, meaning it did not learn enough from it and will not be able to reproduce accurate predictions on new samples, which leads to poor performances. On the opposite end, if the model is trained over too many epochs, it could show some **over-fitting** effects. This unpleasant phenomenon occurs when the model learned the data by heart and it is unable to generalize to new observations, meaning its testing performance will decrease dramatically compared to the training one. The number of training epochs is therefore a important parameter to define.

2.2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) are based on the same principal as the "basic" Neural Networks. However, they contain a major difference with the former network. A **RNN** has a recurrent connection to itself. It is represented by a connection pointing to itself (like a loop). A set of edges goes from the first hidden layer to itself. Like that, the next set of input signals will be combined with past information (left on Figure 2.6). A good intuitive way to picture the recursion would be to "unfold" the model (right on Figure 2.6). The set of connection from the hidden layer points towards the first hidden layer of the next iteration's model. The hidden connection behaviour is defined in Equation 2.8.

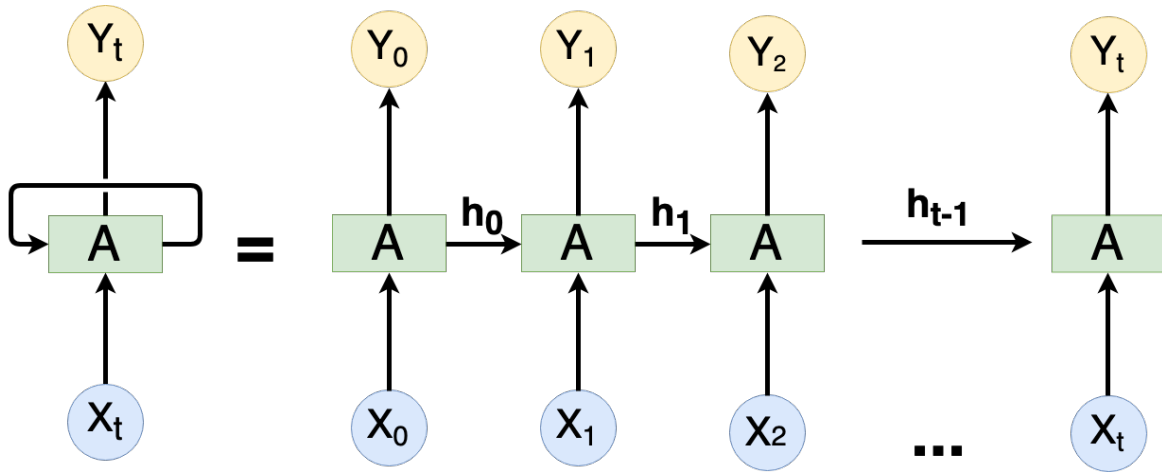


Figure 2.6: Recurrent Neural Network Representations

This particular structure enables a feedback transmission, iteration after iteration, from the hidden layer to itself. It allows the network to have additional information regarding the "behaviour" of preceding weights. This connection from past is what makes the Recurrent Neural Network develop some extra capabilities of recall and recognition, mimicking some part of the human brain intelligence. This transformation of the cells' connection is not without cost and is provided by the "evolution" of the artificial neuron, called a recurrent neuron.

Recurrent Unit Besides the weights, representing *knowledge*, the neurons actually need *memory* to integrate the past observations into their future predictions. The recurrent unit [38], which recurrence equations are stated below, is the modified version of the artificial neuron.

$$h_t = g(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (2.8)$$

In Equation (2.8), x_t is the input signal at time t , h_{t-1} represents the information memorized at time $t - 1$ and b_h is the hidden bias vector of the current layer. The matrices W_{xh} and W_{hh} contain the weights of the respective layers. This operation does not correspond to the summing function operation of the classical neural network, it is happening before it and makes the particularity of the recurrent unit. Similarly

to all linear combinations in neural network, the result is pre-processed by a hidden activation function, g , to limit its amplitude. This summation term is then passed to a second activation function, see Equation (2.9), where it is combine with another weighting matrix and a possible bias. The outcome of this operation is equivalent to the process happening in Equation (2.2).

$$z_t = \rho(W_{hz}h_t + b_z) \quad (2.9)$$

Otherwise, this type of neural network works the same way as its parent network with one notable difference. Due to supplementary feedback loop, caused by the recurrent unit, the back-propagation of the errors will behave in a different manner. In a basic neural network, each weight of each neuron is updated thanks to the propagation of the errors. In a recurrent network, neurons are connected to neurons from a previous layer, but they are also "connected" to neurons of the same layer many iterations back in time, because of the inner feedback of each layer. In that sense, the propagation of the errors need to be applied all the way back to all neurons helping to predict the current output. As the propagation hops, back in time, from one neuron to another, the same set of weights, W_{hh} , is multiplied over and over. In any case, whether the weights are small or large, the **vanishing** or the **exploding** gradient problem arises, respectively. To remedy to the issue, some improvements have been brought to the recurrent unit, one of them is the *Long-Short term Memory* unit.

Long Short Term Memory

Solving the vanishing gradient problem [39], *Long-Short Term Memory* cell has been, firstly, developed, by Hochreiter and al., through an analysis on how the error vanishes as it gets propagated back [40]. This new type of artificial cell provides meaningful insight about the memory's challenges that the network faces as it is presented with newer samples. It is one of the most popular solution to that problem [41], with another type that is a closely related variant, called the *Gated Recurrent Unit*, that is not addressed during this theoretical review, because it showed equivalent prior results.

On Figure 2.7 is represented the inner structure of a LSTM cell, several entry gates and exit gates are shown, they all have a different purpose for the functioning of the cell. On Figure 2.8, the equations applied in the schema are demonstrated. The following explanation links the content of the two figures together. The key idea of this cell is the cell *State*, represented with the letter c in Equation (2.13) or by the horizontal line running in the top part of the cell, in Figure 2.7. Thanks to this state, the information is safely transmitted from entry to exit. Along the way, information flowing through this state can be updated and regulated with three different gates: the **forget**, the **input** and the **output** gates, each one having a specific purpose [42].

Forget Gate : It is the first of the three gate processing the incoming data, x_t the new input and h_{t-1} the information from the previous state. For each pair of values within the (x_t, h_{t-1}) vectors, this gate decides whether or not to keep this linear combination, see Equation (2.10), further in the cell or to forget it, therefore the name. The outcome of this operation is a new vector with some information kept and others forgotten, f_t .

Input Gate : This second gate acts like the one adding new information to the current state. It is composed of two different steps. First, with Equation (2.11), it decides which values will be updated. Then, it computes, with a activation function $g = \tanh$, a vector of new candidate values (Equation (2.12)). These three intermediate results are, subsequently, used to update the current control vector, c_t in Equation (2.13), using convolution operation between the different terms. The control vector is, primarily, revised by the outcome of the forget gate (first term of the equation). Next in order, comes the update of the new value that are added to the flow (second term).

Output Gate : Finally, the third gate decides which values of the cell state will be sent to the next cell. As before, a linear combination is passed through a *sigmoid* activation function, creating a vector of 0 and 1, o_t , controlling which values of the state h_t will be outputted for the current cell result, see Equation (2.14). The weights transferred to the next cell's state are, lastly, a multiplication of the current output, o_t , and a hyperbolic tangent transformation of the cell control's values, see Equation (2.15). This new state vector h_t represent the past knowledge "advising" the future cells states.

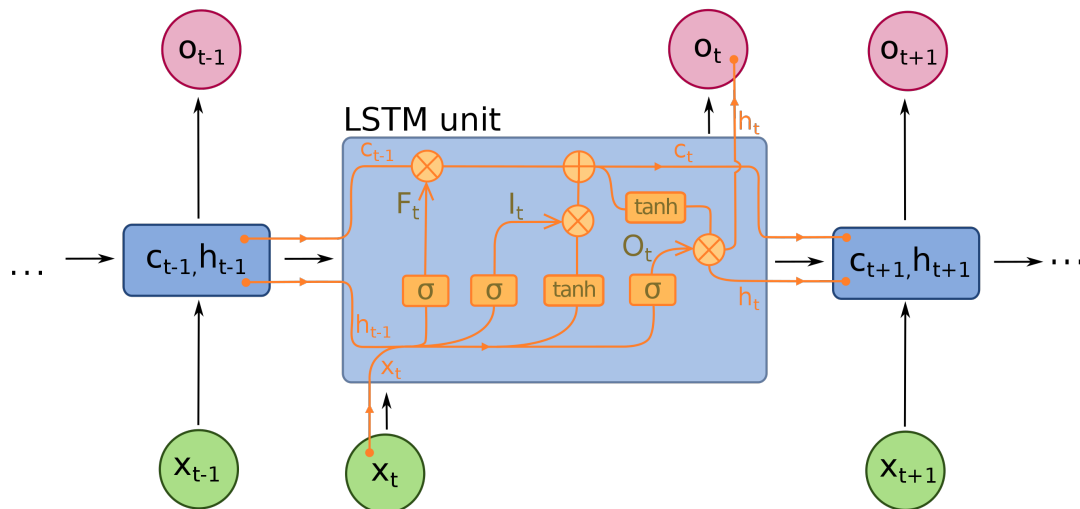


Figure 2.7: LSTM Representation

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (2.10)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (2.13)$$

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (2.11)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (2.14)$$

$$g_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (2.12)$$

$$h_t = o_t \odot \tanh(c_t) \quad (2.15)$$

Figure 2.8: Recurrent Equations following the order in which they are used within the LSTM cell

Brain Analogy As it is the case for Neural Networks, once a Recurrent Neural Network has processed the information forward, a second propagation happens backward. One could use the famous expression, “We learn from our mistakes”, to summarize this process. Once again, Recurrent Neural Networks do this ”learning” process in a similar way as, we, humans use our memory to remember past events and learn from them.

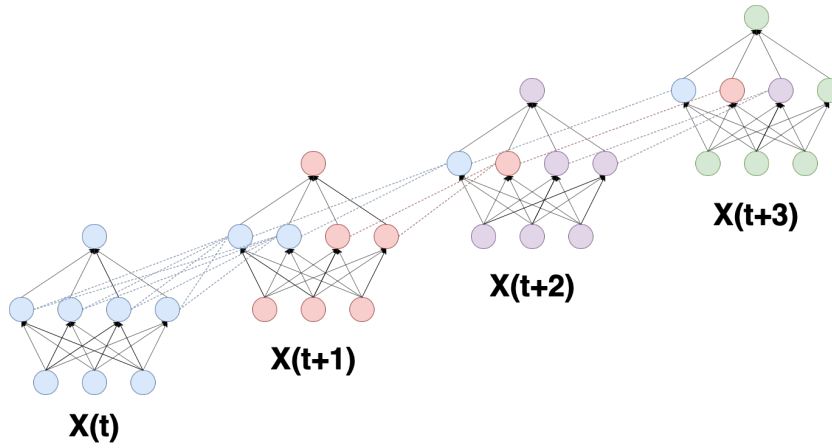
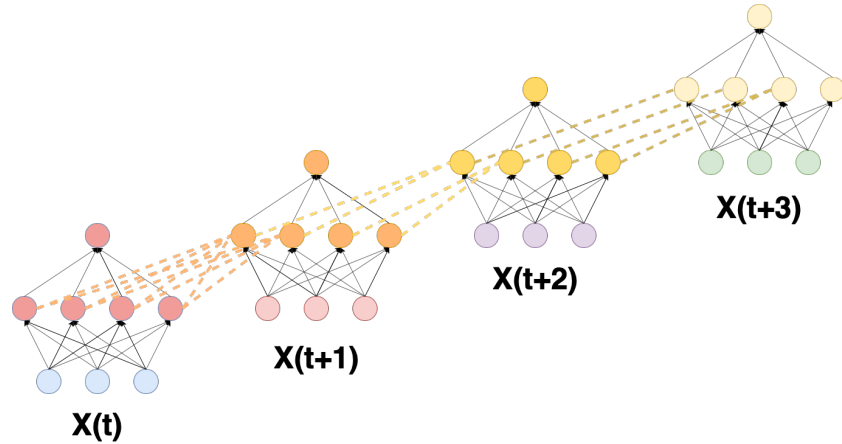


Figure 2.9: Forward propagation of the input values over several time steps. Each color of a node refers to the information passed along the different states over time. Small bits are kept, others are forgotten, as said before.

Above, the detailed process of a RNN processing four different time steps. On figure [2.9](#), one can visualize how the network recalls the past information. Particularly, we can observe that not all the information from previous events is passed along. Indeed, the network, as our brain, retains only the most influential events from the past. As the input signals are fed to the model, each time it keeps a bit of information from every previous state. If more steps had been drawn, one could have spotted that, at one time, the model would have needed to decide which information would have been forgotten or not (for example, the red cell would be replaced by a new information, represented by a orange one). Indeed, after a while, new information are available and they are, probably, more relevant for the more recent states, this is why the **forget process** is necessary. Furthermore, as the human brain’s capabilities are limited, the network is not able to recall every detail of every observation. Moreover, once again, the artificial network, as the brain, only retain significant events and their outcomes so that, when the time comes, it can detect their occurrences and produce the appropriate response, like linking long distance events between each others. However, as one can see on Figure [2.9](#), this storage ability seems to be strongly depends on the number of available neurons within each layer of the network. Experiments about this intrinsic characteristic will be conducted later on during the model building phase in Chapter 4.

Then, on figure [2.10](#), the back-propagation occurs, not only, on the current model’s

Figure 2.10: Back-propagation of the error within the entire network of neurons. The red color panel indicates which neurons are affected by the retro-propagation.



state but on all the previous ones (with respect to the most recent state's output). LSTM cells were specially designed to deal with this phenomenon.

Finally, I would like to conclude about RNN's with one last brain's analogy. As humans, this type of model is able to both perform *Recognition* and *Recall*.

Let's illustrate what are these two fundamental properties of our brain. *Recognition* is the ability to associate a current situation with a previously encountered one. *Recall* is the capacity to remember a fact that is not currently happening (like remembering the name of a person you met last week). A recurrent neural network, thanks to the previously stated characteristics, recurrent layer and LSTM neurons, has the competence to do such "intelligent" behaviours and thus to act like a human brain would do [43]. Combining human-like capabilities with a computation power, no humans can possibly possess, recurrent neural network could be really useful for the task at work, that is, learning meteorological past event over a long period of time to be able to forecast future events.

2.2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are another type of Deep Neural Networks, most often used in Visual Recognition and Image Processing. They differ from the basic neural networks by their more complex structure. This modified structure enables the network to perform an automatic feature selection on data in the form of multiple arrays [44]. Below is an example of a 2D structure (for image processing, for instance). Applied to 1D sequence, like time-series, convolutional neural networks keep the same layout of the network. The only difference is that the convolution and the pooling steps are done, on each variable individually, using vectors rather than matrices to operate the convolution and the pooling processes.

Structure As you can see on the figure [2.11] above, a convolutional network contains multiple successions of convolution and pooling operations. Below is a detailed description of these two types of operations and their associated layers.

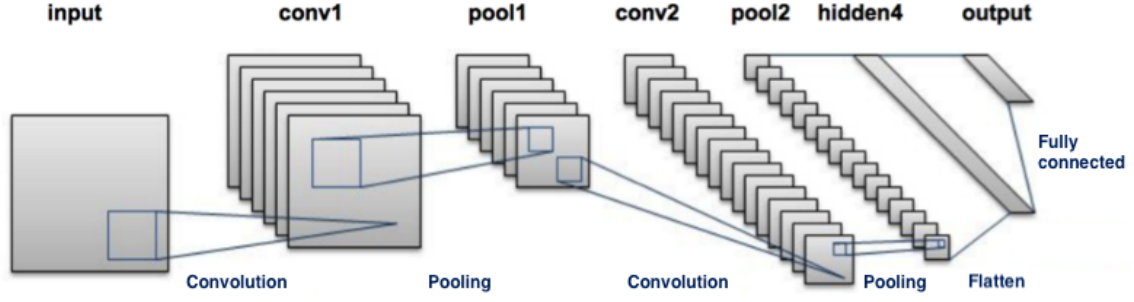


Figure 2.11: Convolutional Neural Network Structure

Convolution : A convolution is a weighted function w , of length $2k+1$, applied to an input signal x , resulting in a new signal s . The weighting function can be seen a fixed size window "sliding" over the input signal x .

$$s(t) = (x * w)(t) = \sum_{i=0}^{2k} x(t - k + i) \cdot w(i) \quad (2.16)$$

In Machine Learning terminology, one would refer to the arguments as followed : x is the **input** (a 2D image or a 1D sequence), w is a **kernel** and s is called a **feature map** [44].

This first stage acts like a filter on the data. It detects patterns within local consecutive sequence (1D) or within the local neighbourhood of a pixel (in 2D), for example. Applying different filters allow the model to produce many feature maps and thus, to represent the data in many different ways. Each feature map outlines an intrinsic characteristic, e.g. a feature, of the data. Now, a better way to visualize this action, imagine the ensemble of feature maps as a light string, each bulb representing a map. When a model gets a new sample, it travels down the light string. Along each bulb, acting as a feature map, the sample sparks a light when the map detects common features between the input and itself. The brighter the light, the higher the resemblance. This brighter light implies that a larger weight is passed to the next layer.

Pooling : This next step is performed on each feature map from the previous layer. The pooling operation consists of replacing the output at a certain location by the summary statistic of the nearby outputs [44]. Consequently, we try to make the dataset more robust to small transformations of the input. In doing so, convolutional model care more about the presence of a pattern rather than its location in space (for 2D image) or in time (for 1D sequence). Different types of pooling exist, like the **Max** (most popular), **Min** or **Average** poolings.

After the pooling operation, the dimensions of the input is decreased by a factor corresponding to the size of the pooling window. For instance, with a 1D time-series, after the convolutional layer, the input's dimension are 60×100 , 60 time observations with 100 feature maps. The pooling window size is 5, the dimensions after the pooling layer are equal to 12×100 .

These two types of layer can be stack up in an infinite number of possibilities with other type of layers, as fully connected ones. Prior to that, the last layer of pooling needs to be flatten, see [2.11](#), to be connected to a dense layer, as dense layer can't deal with 2D inputs.

At this point, the efficiency of convolutional networks on 2D inputs (e.g. computer vision) doesn't need to be proven anymore. However, the characteristics, described above, could provide interesting insights once such models are applied on time-series. The objective is to evaluate the efficiency of CNN's capabilities of feature and pattern detection, compared to RNN's strengths, which reside on the memory capacities.

2.2.4 Convolutional Recurrent Neural Networks

Finally, a more experimental type of model would be to combine the two models previously mentioned into an unique one. Joining a convolutional and a recurrent models into one is done by successively stacking up the key layers of each model. In that sense, a convolutional recurrent neural network would be a structure of convolution and pooling layers, followed by some recurrent layers (LSTM's). The aim of such fusion would be to associate the advantages of both models. Indeed, on paper, the competence of memory paired with an efficient pattern detection would be able to detect patterns on a broader range of times. In practice, we will investigate if it is indeed the case and if such models can methodically discover additional features.

Chapter 3

Data Analysis

During this chapter, we will accomplish the data exploration process. Data exploration is an essential step during every Machine Learning Project. Throughout this analysis, we will get to know more about what constitute the dataset! What are its key characteristics? Are the variables related in anyway between each others? With the help of standart statistical tests or graphical visualizations, the examination of the data will help us understand how the models should be efficiently built. Including more than two millions observations per station, the available dataset counts more than 10 Millions observations that could help achieve an accurate and consistent prediction of the solar irradiance ratio.

3.1 Data Exploration

Stations The observations come from five different stations located on the *La Reunion* island, in the Indian Ocean. On the map, on the right, one can see where each station is situated on the island, along side a table with the lat/lon coordinates. Additional information are present on the map, like the altitude (above the sea level) below each station's marker or else, the red line dividing the island into two parts : the western part is the **leeward side** and the eastern part is the **windward side**. The three western most stations (Possession, Saint-Leu and Saint-Pierre) are thus protected by the high mountains from the strong trade wind blowing from East to West [45]. This important fact might have an impact on the future learning process of the models. Along all chapters, a color code is used so that the readers can easily, and visually, identify the stations within the results' analysis.



Figure 3.1: La Reunion Island

Stations	Latitude	Longitude
Moufia	20.91731	55.476445
Saint Andre	-20.962797	55.622433
Possession	-20.93	55.33
Saint Leu	-21.200642	55.302332
Saint Pierre	-21.313922	55.451069

Table 3.1: A table outlining the 5 stations geographical characteristics.

Furthermore, *La Reunion* is located near the tropic of the Capricorn and its weather is characterized as "tropical", meaning that the temperatures are relatively high (yearly average of 18 Celsius degrees). It might be also important to bear in mind that in the South Hemisphere, the seasons are inverted compared to the North Hemisphere, with summer beginning in December.

These meteorological particularities will probably have an effect on the model learning and prediction tasks, this is why it is decisive to pinpoint these different aspects.

Variables Each station records a set of variables every minute during day and night. The recorded variables (all the same for each station) are the following ones:

Patm : the atmospheric pressure (in hPa).

RH : the relative humidity ratio in the air (in %)

Text : the outside temperature (in Celsius)

WD : the wind direction (in degree)

WS : the speed of the wind (in m/s)

FG : the global solar flux (in $W \cdot m^{-2}$)

FD : the diffuse solar flux (in $W \cdot m^{-2}$)

While the five first variables' meaning are quite clear, the two last variables require some explanations. The solar beams can be categorized into 3 parts: the direct, the diffuse and the reflected radiations. The direct radiations are travelling in a straight line from the sun to the earth (or the sensors). The diffuse radiation is the sunlight scattered by molecules (small particles) in the atmosphere (such as clouds). The reflected radiations are radiations that have been reflected off by non-atmospheric environment (such as ground, lakes, trees). On Figure 3.2, the three different type of radiations are represented. Thus, **FG** corresponds to the global solar radiation received by the sensor and **FD** is the sum of the diffuse and reflected radiations.

For the research question's purpose, the variable of interest is the direct solar irradiance ratio, k_b , computed as the ratio between the direct radiation and the global radiation metrics.

$$k_b = \frac{\text{Direct Radiat}^o}{\text{Global Radiat}^o} = \frac{F_G - F_D}{F_G} \quad (3.1)$$

The range of values for the direct solar ratio spans from **0**, representing a sky covered totally by clouds, to **1**, a perfect sunny day.

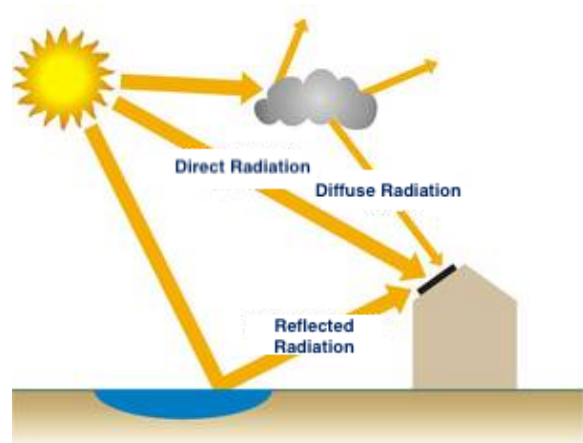


Figure 3.2: Sunlight Characteristics

As said before, these seven primarily variables were recorded each minute during five years, from the 1st January 2013 at 00:00 to the 31st December 2017 at 23:59. For each station, it represents 2,629,440 values by variable, or approximately 18 millions values per station!

Descriptive Statistics

These statistics include the mean, the standard deviation, the minimum and maximum values. This summary of coefficients, per station and per variable, allows the user to understand the tendency and the over all values of the data. By this mean, one can uncover some discrepancies on some variables, like gap in the sequence or values being completely out of range. It is important to note that all the analysis have been conducted on the dataset used for the training, it means the full year 2017 was left out of any analysis. The last year of the dataset is used as the model testing data. If inferences on the structure of the model would be made based on data analysis, including the testing year, would create some biases for the generalization of the model. The end goal is to create a powerful model for any observations, not only for the ones we currently possess.

		Moufia	Posses- sion	Saint Andre	Saint Pierre	Saint Leu
Patm	mean	977.48	1014.39	993.24	1007.54	989.05
	std	3.83	4.31	4.18	4.35	4.11
	min	956.2	992.4	970.18	975.07	954.25
	max	987.7	1025.9	1004.67	1020.12	1000.1
RH	mean	64.8	59.27	69.57	61.82	64.3
	std	9.77	8.98	9.01	9.02	8.62
	min	21.65	24.27	27.1	17.67	27.03
	max	94.9	94.0	96.3	93.8	94.7
Text	mean	23.54	26.46	23.7	25.51	24.2
	std	2.31	2.42	2.44	2.7	2.38
	min	15.0	17.6	14.2	15.0	15.68
	max	31.8	34.48	33.15	34.48	31.58
WD	mean	122.07	173.97	136.86	158.16	193.08
	std	85.6	121.87	71.63	77.9	87.83
	min	0.0	0.0	0.0	0.0	0.0
	max	360.0	360.0	360.0	360.0	360.0
WS	mean	3.82	2.12	3.79	3.84	3.59
	std	2.22	1.09	1.88	1.95	2.21
	min	0.0	0.0	0.0	0.0	0.0
	max	18.25	18.03	21.7	18.07	27.3
k_b	mean	0.42	0.45	0.39	0.54	0.46
	std	0.36	0.34	0.34	0.35	0.34
	min	0.0	0.0	0.0	0.0	0.0
	max	1.0	1.0	0.94	0.94	0.94

Table 3.2: Descriptive Statistics for the five stations.

For each station, the statistics have been conducted on the observations when the sun was up in the sky, reducing the number of observations to 1.3 millions per variable. Indeed, the prediction of the solar irradiance has no interests at night, when the sun is down. Only few of the night time observations will be used, when the first solar ratio of the day will be predicted. The following assumption is made : the early morning, late night, samples are similar to the diurnal samples and they do not significantly change the statistics on Table 3.2

From the above table, the subsequent observations can be inferred :

- The average atmospheric pressures for Possession and Saint Pierre are greater than the others. It is due to the relative closeness of the sea level. An altitude closer to the sea level implies a higher atmospheric pressure.
- Saint Andre's average relative humidity seems a bit higher than the four others. Looking at their respective densities (Figure 3.3), the distribution for Saint Andre is, indeed, shifted by 10% to the right, with a fatter right tail, while the four other station's distribution are equivalent, more or less.
- Concerning the average temperature, Possession and Saint Pierre seems to have higher temperature (by two degrees). Overall, each station has an average temperature, above 18 °C, minima and maxima within the same range, which confirms the tropical climate zone.
- About the wind direction and speed, the statistics are not really revealing. This is why, on Figure 3.4, we can visualize better how the wind directions are distributed throughout the five years. Recall that Moufia (blue) and Saint Andre (green) are **windward** stations, they are directly exposed to winds, normally blowing from the East (the Alizee), about 90 degrees from North direction. Though the distributions are not quite identifiable, on one hand, we observe that wind is usually blowing more from the east and the south [90 to 180 degree], for the windward stations. On the other hand, the leeward stations experience mostly a north to south directed wind, blowing from [315 to 45 degrees], that is somehow, more spread out on the cardinal axis.

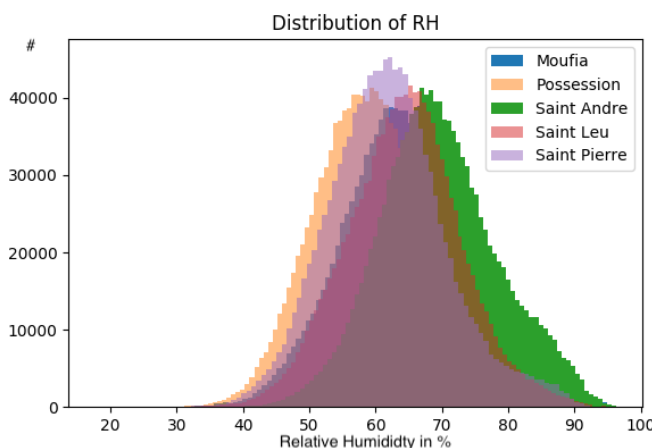
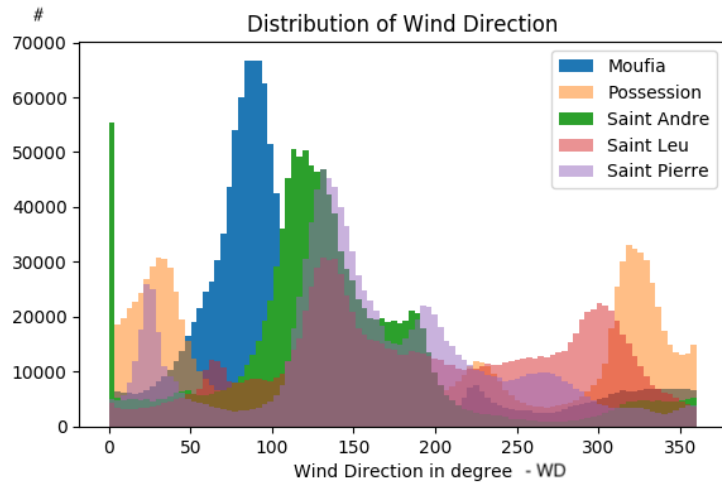


Figure 3.3: Bar chart of the humidity ratio for each station, represented by its dedicated color. The $x - axis$ represents the percentage of humidity in the air while the $y - axis$ is the number of observation with the respective value.

Figure 3.4: Keeping the same color legend for the stations, this bar chart describes the number of occurrences, within each station, of the direction of the wind.



Correlation

As mentioned in Section 2.1 i.e. Time Series Analysis, the correlation metric is a way to determine the degree of likelihood of linear relation between two variables. Across the five different stations, the original day and night variables, so to keep a continuous sequence, have been tested on the existence of a linear relation among each others. On Figure 3.5, a value close to zero indicates no linear relation (neutral). A value near the extremes $(-1, 1)$ indicates a strong linear relation, whether it is inverse proportional or directly proportional.

Without looking at the trivial diagonal values of each correlation matrix, the first two tables of the next page depict the correlation of the windward stations, Moufia (on Figure 3.6a) and Saint Andre (on Figure 3.6b). It shows that the correlation range from -0.35 to 0.25 , which is quite low for a linear relationship. However, the only pair of variables demonstrating a greater linear coefficient, for the windward station, is the Atmospheric pressure - Exterior Temperature couple. From a physics-based point of view, this last observation makes sense, the higher the temperature, the greater the pressure. In closed environment, this is caused by the greater energy, thus greater moving speed, each air molecule possesses due to the heat. Nonetheless, in an open space, the physics rule doesn't apply. The data indicates that the relation is inversely proportional. In such environment, other factors such as the location and the associated micro-climates, e.g. **rain season**, can disturb the natural behaviour of these two variables. This demonstrates that our intuition cannot always be trusted and that deeper analysis reveal interesting features.

Analyzing the relations between the leeward stations' variables, one detects almost the same observations; the input variables are lowly correlated between each others and with the target variable, the solar ratio. This near-zero correlation statistics between the input variable and target indicate it doesn't exist a linear relation between them. Gladly, neural networks are capable of modelling linear relations and also non-linear relations. If non-linear relations exist within the present dataset, the studied models have the capacities to detect them and create models to tackle the relation-finding task, as seen in section 2.2.

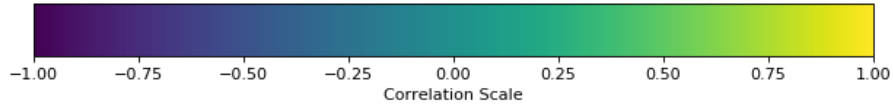


Figure 3.5: Scale for Correlation

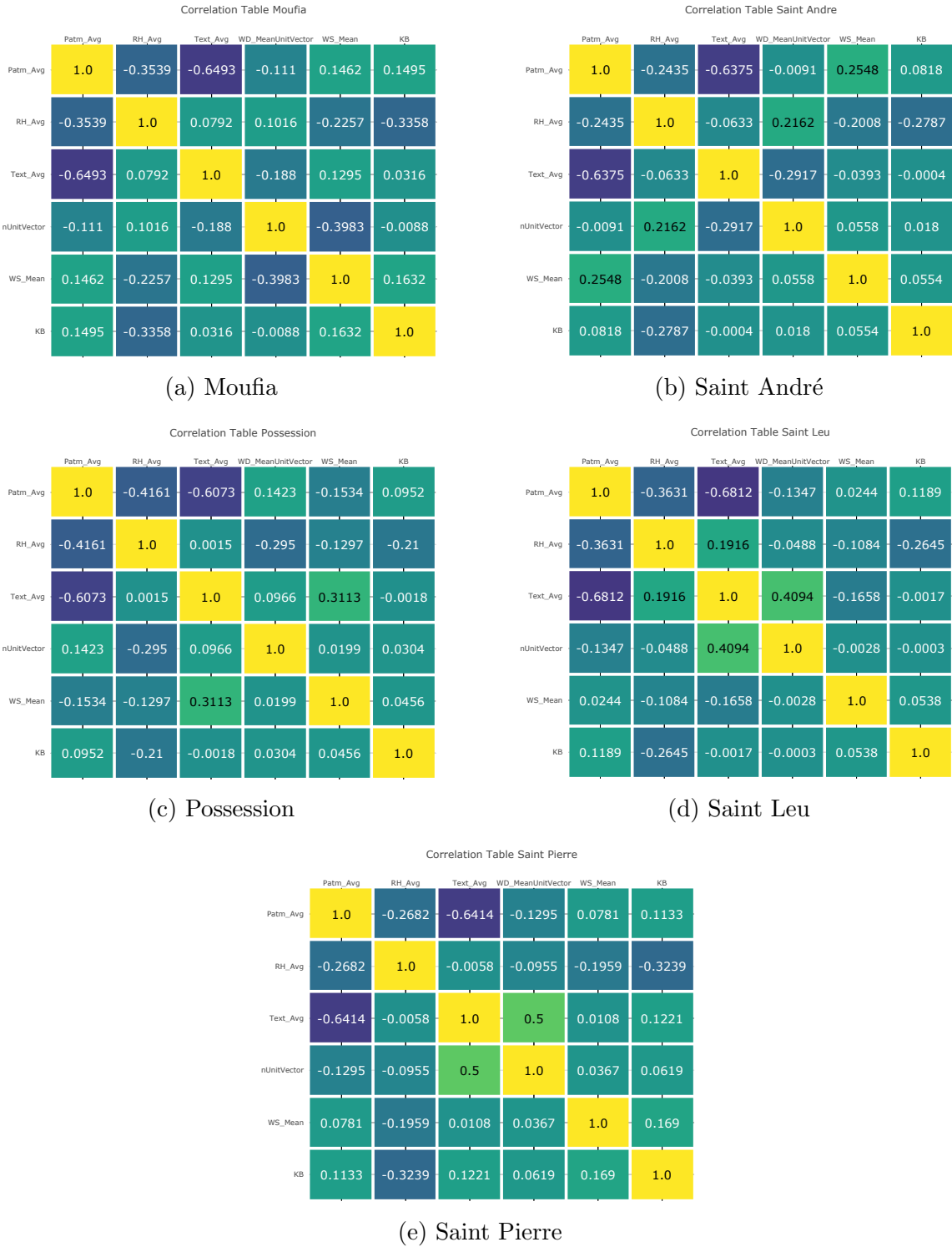


Figure 3.6: Correlation Tables

Correlation Inter Station

Considering the five stations and all possible correlation pairs of the same variable from two different stations, correlation distribution across the stations could be established for each variable. Like for every distribution, the mean and standard deviation are good indicators on the form of the distribution and how strong could the linear relationship be. The results from 3 types of combinations are displayed on Table 3.3. **All stations** contains the pairs of all the possible combinations. **Leeward** category contains only the pairs of the combination when both stations are from the leeward category (e.g. *Possession*, *Saint-Leu* and *Saint-Pierre*). The last row, **Windward** contains only the distribution characteristics from the 2 windward stations combinations. This will be useful when we will build the spatio models in the fourth chapter.

Category	Patm	Rh	Text
All stations	0.991 ± 0.004	0.562 ± 0.089	0.878 ± 0.027
Leeward	0.991 ± 0.004	0.555 ± 0.073	0.878 ± 0.026
Windward	0.990 ± 0.004	0.572 ± 0.113	0.878 ± 0.030

	Wind Dir.	Wind Speed	K_b
All stations	0.017 ± 0.191	0.228 ± 0.166	0.423 ± 0.088
Leeward	0.0290 ± 0.200	0.198 ± 0.148	0.421 ± 0.076
Windward	-0.001 ± 0.187	0.274 ± 0.191	0.426 ± 0.110

Table 3.3: Correlation between sites

Naturally in light of the relative small size of the La Reunion island, the atmospheric pressure and the temperature are highly correlated between stations. The rest of the table shows a lot of heterogeneity among the stations. While combining the data and using it through out classical models seems like a waste of time, in a later chapter, we will investigate the possibilities to build a spatial model to exploit all data based on their location in space. This could be particularly interesting noting that the variable distributions related to the wind direction and speed differ once the distributions are isolated by the wind category.

Auto-correlation

More over, one last aspect of the correlation study is the auto-correlation within variables. Figure 3.7 shows the auto-correlation among the six different variables for Saint-Pierre station. As for the inter-station correlation, one can see that the atmospheric pressure and temperature follow a natural behavior, e.g. cyclostationarity. This process is defined as being periodic in time. Indeed, looking at the blue line on the graph, we recognize a recurrent pattern with, approximately, a one year period, but it doesn't mean that value are exactly the same as the year before. It only indicates they have a linear relation along the years.

Nevertheless, all variables don't follow a cyclostationary process. Besides the atmospheric pressure and the temperature, none seem to have a significant recurrence re-

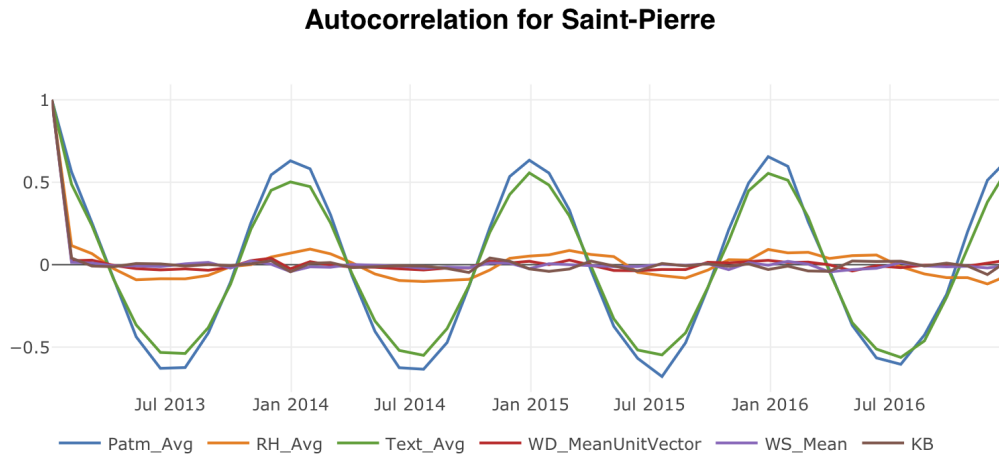


Figure 3.7: Auto-correlation

lationship. Indeed, the wind speed and direction and the solar irradiance ratio don't naturally follow a yearly pattern, or smaller time correlation at all, as they probably depend on other local factors!

Conclusion Being simple but effective, these descriptive statistics give us more background knowledge in order to understand the data we are using. With a bit more computation, we can also discover how the data is distributed and if any trend lies in the sequence of observations. Keeping in mind what was stated during the **time-series** introduction, it is indeed beneficial to plot the data, whether it is in a line chart or a bar chart, to figure out how the data behave during these 5 years, as on Figures [3.3](#) or [3.4](#).

Thanks to these multiple analysis, one question arises : the observations don't show clearly defined pattern or features over this five years period, will the Recurrent Neural Network be as efficient as it is assumed, when applied to other data fields ? On the same topic, couldn't the Convolutional Neural Networks be as competitive as the RNN, seeing how some features, like (auto) correlation, are strongly defined among the variables ? From this aspect, expectations on the performance of both types of networks are equivalent !

3.2 Pre-processing

All the preliminary analysis have been done on raw dataset, which mean the data was processed as is. However, to build a data that will, hopefully, allow the models to perform efficiently, the dataset needs some pre-processing tasks. Pre-processing is another crucial step of a Machine Learning project, it defines the quality of the data and the usefulness of the information. It is crucial as, without it, the ability of a model to learn can be greatly affected. Operations like values' cleaning (detecting absurd values), normalization or standardization (scaling the values within a specific

rage) or dimensionality reduction (lowering the number of variables), to eventually visualize the data, will be applied on the raw data to form a new dataset.

Monthly Hourly Normalization When dealing with meteorological data, we have to take into account (strong) yearly and daily cycles in the data [46]. To do so, the "monthly-hourly normalization" is considered and applied as followed on the dataset:

$$x_t^{norm} = \frac{x_t - \text{mean}(x_{m,h})}{\text{standard deviation}(x_{m,h})} \quad (3.2)$$

where $x_{m,h}$ is a set of data defined by $x_{m,h} = \{x_t \mid \text{month}(t) = m \wedge \text{hour}(t) = h\}$. Applying this transformation, it resulted in a weak stationarity dataset where the mean and standard deviation are, approximately, constant for $\forall t$. Consequently to this "rule of thumb", the RNN and CNN models could have some difficulties to detect important pattern or, to remember events through out the years. This is why, in further sections, it will be important to monitor how the models perform on data that has been monthly-hourly normalized or not. A comparison between the models trained on the two datasets, a raw one and a Monthly-Hourly normalized one, will be conducted and checked for any notable differences.

MinMax Normalization Posterior to the first normalization, which doesn't scale down the variables' values within a range, the *Min-Max* normalization aims at changing the numeric values of the dataset to a common scale, between 0 and 1. This pre-processing step is really important as unnormalized dataset, with different variable's ranges, might have longer training phase, where the gradients of the back-propagation are really slow to converge (or even no convergence at all) [47]. More over, unnormalized dataset could contain variables, which ranges are disproportionate compared to one another. The resulting effect would be to create unbalanced models as the variables with larger and greater ranges could shadow the importance of variables with smaller and lower ranges, but still conveying very important informations ! Nevertheless, one need to be careful when applying this normalization. If a variable's distribution can not be roughly defined as a Normal distribution, this transformation will do more harm than good to the behaviour of the resulting values!

Dimensionality Reduction A third pre-processing step that is the reduction of variables dimensions could be investigated. This action, of reducing the number of variables, has 2 main goals : firstly, to reduce the load of input data for the model, specially when dealing with the models of interest and secondly, reducing the number of variables up to 2 or 3 dimensions might be a good solution to visualize and/or analyze it on graphs. Different methods exist to achieve this goal, such as Principal Component Analysis, Factor Analysis or t-distributed Stochastic Neighbor Embedding. These methods operate in such way that they try to keep (and/or transform) the most meaningful variables into new variables, in regarding of criterion such as the variance within samples or the proximity of samples. This is why considering this approach might shed a light on possible improvement! However, after some preliminary tests, the results weren't as conclusive as expected. Additionally to this fact and the small number of variables, the dataset was left untouched to continue the research.

3.3 Additional Variables

Although the dataset already possessed very interesting variables for this five years period, it exists many other relevant variables available and maybe, some of them would be relevant to the research question purpose. The main difficulty would be to find data with a 1-minute sampling ... during five years! Thus, the challenge was to find generic meteorological data such as sunrise and sunset, or season indicator.

Season Indicator : Simply, based on the day of year, a label between 0 (for summer) and 3 (for autumn) was assigned to each observation. Due to the location of the *La Reunion* island in the **South** Hemisphere, the seasons are switched compared to the North Hemisphere ! This variable will, hopefully, act as an additional feature to help the models differentiate the observations based on the seasons.

Night Indicator : Based on python package *Pysolar*, allowing the extraction of the sun position in the sky based on a localization, sunrises and sunsets for the stations were established. In order to simplify this step, and assuming the island was small enough to generalize this information, one pair of (sunrise, sunset) was defined for all five stations per day. The reference point to do such operation is a average of the five positions, located on the center of the island. Another assumption was that the sun rose (or set) when it was more (or less) than 15 degrees up in the sky. This assumption enables the users to not take into account the special topography of the island. Thus, in some stations, even though the sun rose (or set), the station might not receive light (or still receive light). However, the margin of 15 degrees was defined so that it is rarely the case. Doing so, each observation of every station, has been binary labeled as being part of the night or part of the day. This simple labeling have allowed a pre-classification of the observations into two categories and so, the deletion of the night-time observations as they are not relevant for solar irradiance prediction.

Daytime : This additional variable is based on the time of the day. For each observation, the time, associated to it, was converted into a indicator ranging from zero to one. The aim of this variable is once again supposed to help the models to better map the observation into the **time dimension**.

With only three additional variables, the dataset was increased by almost 50%. However, the expectations of finding more relevant variables were stopped by technical challenges. For example, another variable like the aggregated *Rainfall* (since the beginning of the day), could have been useful to assess the situation of the day. But the measurements weren't consistent enough to fit with the current available dataset, because the sensors, recording the informations, weren't accurate enough or they were

offline for long period of time within the five years window. Concerning the non-meteorological variables, time related variable like the day of the year or the month of the year could be added to the dataset, but the objective was that the models would learn this feature on their own.

3.4 Dataset construction

Due to the models' implementation's specificities, the raw data cannot be directly inputted to the model as is. The dataset needs some transformations in order to be usable. In line with the theory, the *Keras* package [48] is one of the most suitable to put Section 2.2 into practice. However, the theoretical models differ from the practical ones in the way they accept the data. In the theoretical RNN, the observations, x_t , are fed to model one by one. Then, each sample is processed and the "memory" weights, associated to this event, are adjusted and looped to the model's state of the next input, see figure 2.6. For the practical model, each input sample needs to be a block of Δp time-steps, e.g. observations. Δp defines the number of past observations one want to consider for the prediction of the target, its default value is 60, corresponding to an hour of samples. Δt defines the interval between 2 predictions, its default value, for now is $\Delta t = 1$ minute. It is important to note that Δp does not represent the only knowledge to the model, the recurrent capabilities of the models intervene also.

Yet, as previously said, it is not practical for the present work purposes to predict the solar irradiance at night! Nonetheless, nocturnal observations will be included in some input regarding the early morning predictions. Indeed, once the sun has just risen, the model bases its prediction y_t on observations ranging from $[x_{t-\Delta p}, x_{t-h}]$. Consequently for a period of $\Delta p + h$ predictions, the input data include a mix night-time and day-time samples. After this "buffer" period, the input x only contains diurnal observations. This transformation particularity has 2 important effects on the dataset, related to the value of the sum $\Delta p + h$. Firstly, the dataset size could be reduced if this sum becomes greater than the number of samples during the first night of the dataset, the first input could not have a constant size, introducing some implementation problems. Secondly, for some days in the dataset, exceeding this sum will introduce some daytime values from the previous day while for some other days, it will not. This tricky situation could mess with the values of the values at the given input samples. This phenomenon has to do with the variable night length through out the year and also need to be carefully planned. For now, the default value of these two parameters are equivalent to 60 minutes each. The target value is evaluated on one hour of data and is predicted one hour after the most recent observation in the input dataset. It should not cause any problems. If these parameters were to increase, they would need to be carefully monitored.

Looking more specifically at the content of each newly created sample, $\mathbf{X}$ contains n variables including the target of the past values (since they are known). This gives a great amount information about how was the weather in the last hour. Considering that each sample in the formed dataset has p observations, the p^{th} observation has the time index t , last available row for this sample. Thus, this sample target is y_{t+h} if we want to predict the target's value at an horizon h . A visualization representation, on Figure 3.8, is given below to help the readers quickly understand the transformation.

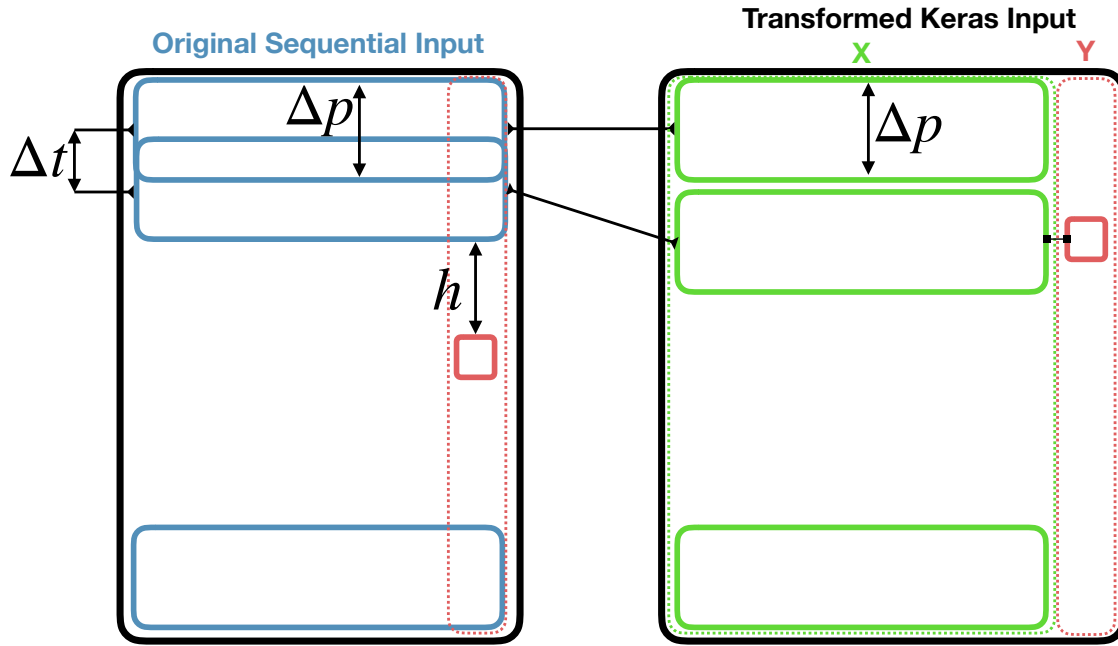


Figure 3.8: Dataset Transformation

3.5 Discussions

Knowing what are the inputs, let's take a deeper look at the value of interest, k_b , the solar irradiance ratio variable. As mentioned earlier, the solar irradiance is a key component in the renewable energy production industry. Determining its future values could improve the planning and operation of the power grid producing electricity from sunlight. In the following subsections, several aspects of the target value are studied and analyzed, such as its distribution over time, the imbalance between the classes or the importance of the "one hour" horizon prediction and their possible variants.

3.5.1 Target Distribution

A first interesting way to visualize the distribution of the solar ratio consists of a **calendar map**. For each day, the average ratio is computed thanks to all the daylight samples within that day. It results in a calendar with 1826 days, each one of them describing the general solar trend of the day. Looking at [3.9](#), a red case indicates a cloudy day, while a green case expresses a clear sunny day. The x-axis represents the months while the left y-axis depicts the five consecutive years and the right y-axis indicates the weekdays.

Red cases are more often spotted for the two **windward** stations, that are Moufia and Saint-Andre. The "sunnier" station of all is clearly the leeward station Saint-Pierre, potentially more protected by the natural cloud barrier that constitute the volcanic mountains. But training a model on one of the three previously mentioned stations would have a bad impact for generalization, like the **cross-site** performance assessment, that is the prediction of the target on some station while the model was trained on a different station. This is why, stations like Possession or Saint-Leu seem more fitted for this task.

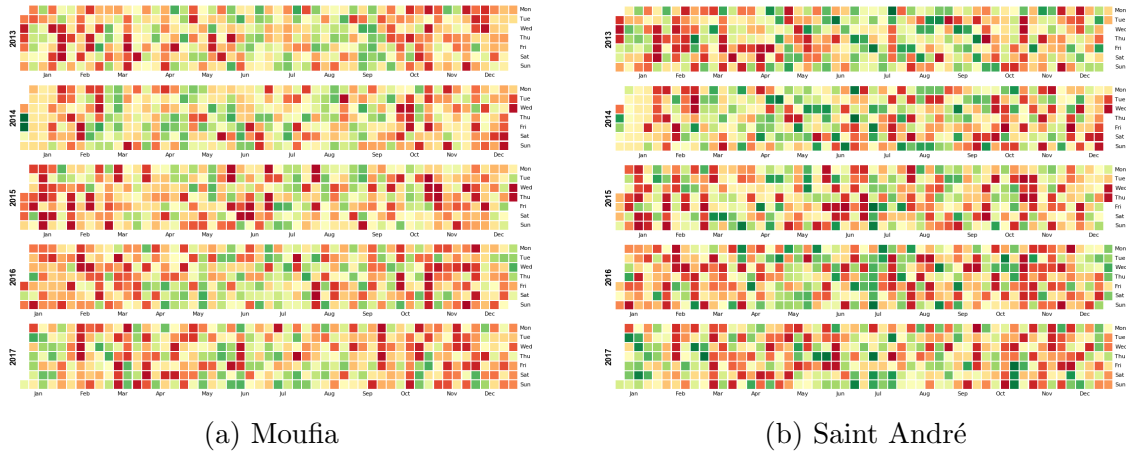
For the two **leeward** stations, *Possession* and *Saint-Leu*, the rainy season occurring during the first three months of the year, is clearly separated from the dry season. For the others stations, the **windward** are relatively more cloudy, which could imply more rain, and *Saint-Pierre* is less affected by the rain. Another indicator that the Eastern coast is more subject to cloudy weather than the Western coast, on average, the Eastern coast of the island receive ten time more rain than the Western coast over the course of a entire year [49].

However, keep in mind that this visualization shows the average ratio day by day. Thus, another important aspect to consider would be the **reactivity** of the model to quick weather variations, that is not shown here. Such characteristics will be assessed during the presentation of the final results.

Related to what was said during the introduction of the section 3.1, in that specific case, the testing year, 2017, was also plotted so to have a global overview of the target distribution.



Figure 3.9: Scale of Kb ratio



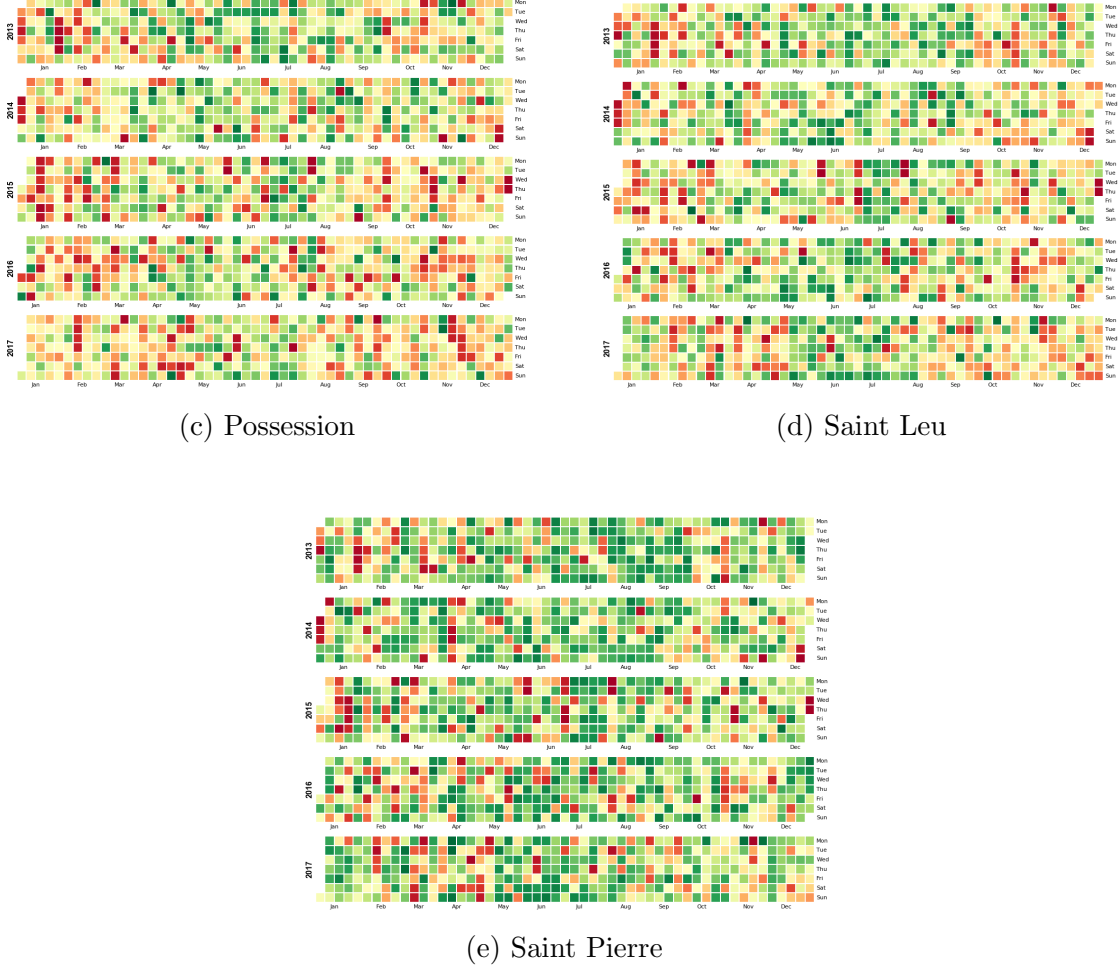


Figure 3.10: Target Distribution by Station

3.5.2 Class Imbalance

During the pre-processing section, all the night values' target were replaced by samples from a specific distribution in order to alleviate the sensors' imprecision during night time. Indeed, some values were not consistent, possibly due to light pollution or malfunctions of the sensors. However, this correction also prohibits the daytime target to be affected by the night time values. Imagine if we would not have removed the nocturnal values, as the ratio varies senselessly across all the interval $[0, 1]$ during this period, the model would have been biased. Because of this bias, the predicted values could lie unwillingly towards one extreme or the other. Such effects would have had huge consequences on the overall performance of the model as observations true distribution would be completely wrong and it would lead to a important imbalance of class.

Gladly, removing the night time values and the target of the testing dataset, the distribution of the target values seems to be, more or less, evenly balanced for the 2 opposite outcomes, for each station. Even though, for some station, the tendency toward one end seems more pronounced, it does not appear to be significant. After a deeper inspection of the graph, one can notice the peak surging on the left end side. It occurs for the two **windward** stations, *Moufia* and *Saint André*. It isn't surprising as the Figures [3.10a](#) and [3.10b](#) show more red squares on this particular chart. The

explanation has surely something to do with the micro-climates present on the island.

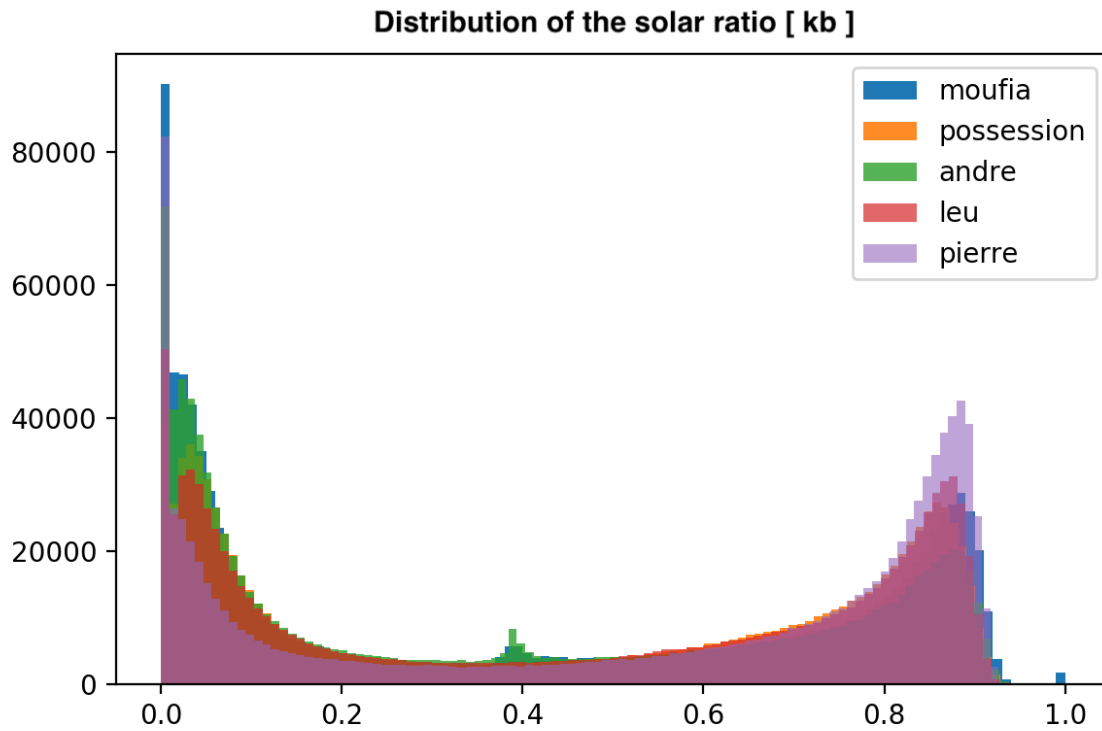


Figure 3.11: Imbalance of the target

3.5.3 Past and future knowledge

Now comes the question of the horizon period. What should be the horizon time prediction ? Well, the perfect delay between the available data and the desired data depends on the task one wants to optimize. Whether it is a few seconds prediction for stability, a few minutes to a few hours prediction for resources and load management or a next day prediction for scheduling, many possibilities exist. Concerning the research question, we are dealing with a 60 minutes prediction horizon, to optimize the load and storage management of the solar electricity produced. An interesting comparison between the performance of different horizon predictions could be investigate during this study.

The research question investigate the behaviour of prediction models based on the last hour of available data. An interesting insight would be to vary the amount of previous observations taken into consideration. Would a larger knowledge imply a better prediction result ? Would a smaller horizon imply a better reactivity or a larger one a better stability ? Such experiences would have been relevant to do. However, due to the important model's training time and the number of model to test for the main research question, these experiences could not be conducted successfully.

Chapter 4

One-hour Prediction Task

This fourth chapter will actually tackle the research question, now that the data is well understood. *"How efficiently can a Deep Neural Network make a prediction one hour into the future?"*. To do so, the first section will briefly introduce how the models are implemented and how the available data was split and evaluated to respect the Machine learning guidelines of the training and testing phases. Then, the second section details the results of the models explained in the theoretical chapter.

4.1 Methods

As there are many different programming language to implement the theoretical model into practical ones or many different methods available to assess these practical models, *Python* [50] programming language was the chosen language as it is a well-known language in the Machine Learning field. In particular, the *Keras* [48] interface is a practical front-end package using *Tensorflow* as a back-end to implement and run deep neural networks models in *Python*.

Regarding the learning procedure, the dataset was divided in a 3-part process where the data is divided into a training, a validation and a test set. Further details on the dataset division will be presented in Section [4.1.2](#).

4.1.1 Models

Each model, whether it is a RNN or a CNN is composed of layers. Each layer can be tuned upon different parameters like the number of "neurons" (the type of the neuron differs within each type of model), the activation function, the connection between the layers, etc, ... Additionally to this important tuning capacity, an infinite number of layer's structure can be derived in the literature and thus create a huge number of models to test! The best way to find the best performing model would thus be to try every possible combination of these parameters. However, as the time is limited, the number of maximum layer was reduced to four, which seems significant enough to create Deep Neural Networks, with thousands and thousands of weights. On the contrary, some parameters were arbitrarily fixed, after some prior comparison tests, to limit the models validation process. For instance, the *Adam* optimizer directly showed greater odds of good performance against the *RMSPprop* optimizer. Indeed, the first

one is an adaptive combination of the second one and adds the *Momentum* technique, which accelerates the search toward the direction of the minima, with an additional term, as seen in the chapter two.

4.1.2 Training and test phases

For each station, the data has been collected from the 01/01/2013 to the 31/12/2017, with a 1-minute sampling. It represents 1,232,768 samples by variable, excluding the night time values. As stated in chapter two, a model learns when fed with data. However, many different learning methods exist, and for this specific time series, the selected technique used is illustrated by the figure 4.1. The five years interval is divided into three years for the training set (used to train the model), one year for the validation set (to choose the best model) and one year for the testing set (to assess the performance on new observations), having for proportions [60%, 20%, 20%] out of the total dataset. However, the separation was not done sequentially, i.e. from 2013 to 2015 for the train, 2016 and 2017 for the validation and test sets, respectively. Instead, to get rid of some kind of correlation between consecutive observations during the training process, the training and validation parts were shuffled.

The "train-val" set spans from 2013 to 2016 and the test set includes all observations from 2017. Yet, the "train-val" set has been shuffled. As you can see in the zoomed time-line on figure 4.1, all the observations have been mixed **by day**. Doing so, the continuity of the dataset is partially "broken", the training-validation set, now, contains bits of samples from four years, which gives more informations about the dataset and enables a better performance assessment of the models during the validation.

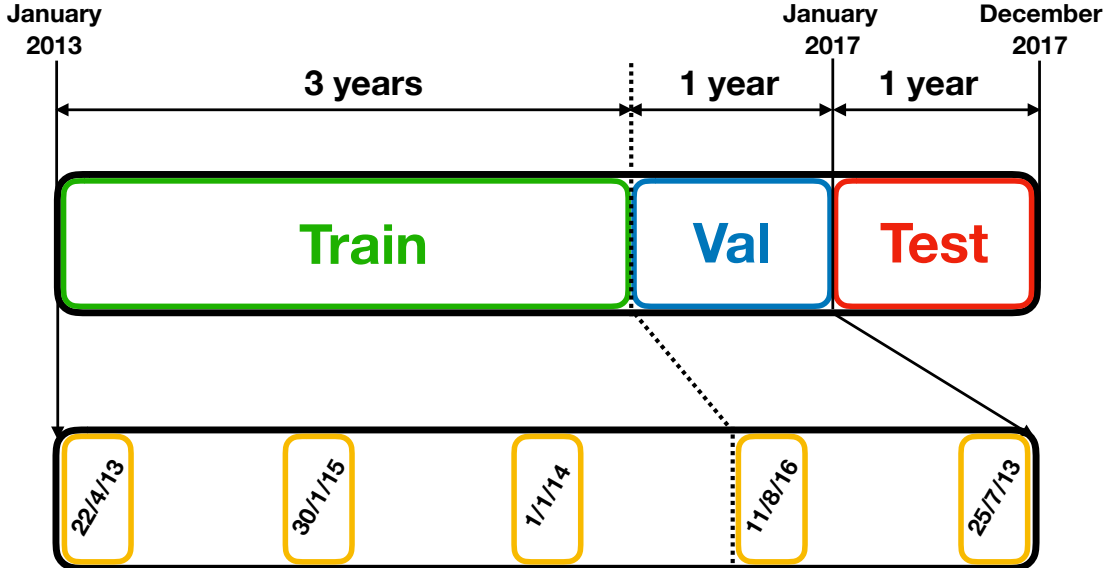


Figure 4.1: Dataset Separation

An additional shuffle has been conducted on the training set only. It blends all the samples between them. Now, the successive training examples are not from the same day, not even from the same hour. A sample from 9:14 can follow a sample from

15:47. The purpose of this "second-level" shuffling is that consecutive samples are not correlated anymore. This operation is necessary as the **optimizers** used during this work assume the samples are independently and identically distributed, which was not the case before this step. Without this action, the models could have been once again biased and led to lower results than expected [51].

4.1.3 Performance metrics

For the cost functions and performance assessment of the models, two well-known metrics, in the Machine Learning field, were employed : the *Mean Absolute Error*, aka **MAE**, and the *Root Mean Squared Error*, aka **RMSE**. The mean absolute error is computed according to the Equation (4.1). It is the average, over all tested values, of the absolute difference between the true and the predicted targets.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (4.1)$$

The root mean squared error score is computed with the Equation (4.2). It is the squared root of the average squared difference between the true and the predicted values.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (4.2)$$

Advantages and Drawbacks So what differentiate the two metrics ? First, both metrics are non-negative, their range span from 0 to ∞ . However, in the present case, the predicted target ranges from 0 to 1. Then, the metrics will operate on the same interval. Indeed, at best, no errors occur (very unlikely) and it results in a 0% of errors. Or, at worst, they are all really wrong and the largest error, that can happen, is equal to 1. Squared or not, and then averaged by the number of samples, the maximum error will equal 1. Thanks to this, these two metrics represent the percentage of samples with a incorrect prediction. As the metrics are non-negatively oriented scores, the models' performances are considered better when their respective metrics are closer to 0.

The main difference between the two metrics is that the **RMSE** squares each error. Hence, large errors have a greater effect on the performance and the metric is useful when large errors are undesirable for the final prediction sequence [52]. However, using the **MAE** will definitively be more robust to outliers.

4.1.4 Parallel and Cross site

Thanks to the spatial dimension of the dataset, several models' insights can be conducted on the datasets. Two different ways were experimented on during the performance evaluation of the models. After training the model with the data from station **A**, it was evaluated on the data from the same station **A**, called **parallel-site** testing.

The second evaluation was called **cross-site** testing. It consisted on training the model on station **A**'s data and then, to measure its performance on the data from the four others models.

The first method assesses the performance of the model on its "own" data, which seems logical. The second method investigates a potential link between the stations' data. While a model has not been trained on the same station it has been tested, it might be interesting to understand how the model behaves, whether it is good or not.

4.2 Results

The main dataset applied to the models is the dataset containing the original and the additional variables, summing up the count to ten input variables (including the historical solar ratio, as it is known), which is "Monthly-Hourly" normalized and then "Min-Max" normalized. Throughout the different trials, the models were also tested on the dataset containing only the original variables and/or not normalized (in any way). If results are better for these datasets, they will be displayed.

4.2.1 Recurrent Neural Networks

Due to the structure's complexity of the Recurrent Neural Networks and to the load of data to process during the training, more than 10 gigabytes of data, the learning process had to be "shorten", otherwise it would have taken way to much time ... Indeed, it exists an infinite number of possible structure settings for Recurrent Neural Networks through the different values of the following parameters : number of normal and recurrent layers (from 1 to 750), number of neurons for each layer , the activation function of each layer, etc, ... However, to reduce the training time of each model and to test more combinations, the value of the **batch size** parameter was, arbitrarily, fixed to a computationally convenient value of 64. This parameter defines the number of samples the network processes before updating its weights with the back-propagated errors. Here, it roughly means the network's weights are updated every hour. Nonetheless, it is not a final value for the batch size, as a smaller¹ value could lead to better results², but induces a greater training time! This effect will be investigated once a final structure has been defined for the Neural Networks.

Parallel-site The parallel models were trained on the equivalent of three years, sampled randomly from January 2013 to December 2016 and validated on one year worth of data. Each model was trained and validated on the data from the same station. Out of the five stations, parallel tests were conducted only on three : Saint Andre, Possession, Saint Leu. This selection has two objectives : firstly, it is to gain parametrization time and secondly, from a geographical point of view, these three stations were selected to ensure a robust and complete analysis of the spatial dimension.

On table [4.1](#), we can see a complete summary of the performances. The parallel results are highlighted with the respective colored background. Overall, *Possession*

¹not too small either as it could lead to high variance gradient estimation during training.

²<https://stats.stackexchange.com/questions/153531/what-is-batch-size-in-neural-network>

gets the best results out of the three stations. Looking back at Figure 3.10, the former station presents the most homogeneous target distribution. In fact, the distribution between the cloudy (red) and sunny (green) day seems balanced, which does not oblige the model to tend towards an extreme. However, whether it is for *Saint-Leu*, that is slightly sunnier than cloudy, or for *Saint-Andre*, that is cloudier than sunny, the trained models were a bit biased and had troubles to replicate theirs learning on the validation set. The shuffling part of the data is not the concern, as the results were averaged on two different trials (per model), using two different input datasets. The only explanation is that some stations have better learning data than others for such task.

On table 4.1, the last column defines the best performing model for each station and for each rank. The station order is the same as the previous columns of the table. On appendix B, each number present in the column is associated with the corresponding network's structure.

About the models' structure, the models giving the best *mae* contain two layers of recurrent neurons, from 150 to 600 neurons. From the other metric point of view, the *rmse*, the models contain also two recurrent layers with, radically, less neurons, from 5 to 50. However, looking at the other metrics of each structure, the results are all really close. It will nonetheless be interesting to compare the results across the stations to understand which structure performs the best.

Cross Site Given the three previously trained models, they were tested on the four other stations. It is a way of analyzing a model's behaviour on a testing set that differs from the training set. In doing so, the generalization capabilities of each model will be determined. Once again, each station shows a top 3 ranking of models, in which the best result is highlighted in bold.

Please note that the *mae* and *rmse* results, within a cell and across stations, don't come from the same model. For each specific case, the best model's result is displayed. For each station and for each metric, the top 3 ranking is independent. The reason behind this presentation of results is to globally observe which models are more efficient.

Surprisingly, for each model, the best *mae* result is obtained when testing the model on another station's dataset, as it is the case for testing Possession's model on Moufia's testing data and Saint-Leu's model on Saint-Pierre's data. However, it is not abnormal either : Saint-Leu and Saint-Pierre, besides the fact of their geographical proximity, both are **leeward** stations. Despite Possession and Moufia are from two different categories, respectively, leeward and windward, they are really close to each other, which could imply similar input data and somehow better results, possibly due to some over-fitting on *Possession*'s data. Finally, due to Saint-Andre's particularities (easternmost station, exposed to the wind), the respective model performs the best on its own data cause the data differs strongly from any other stations and the model cannot find similar pattern in the tested dataset.

		Possession	Saint Leu	Saint Andre	Moufia	Saint Pierre	Models
1	mae	0.1722	0.1921	0.1981	0.1692	0.1737	8 30 29 31 45
	rmse	0.213	0.2374	0.2451	0.2077	0.212	44 30 44 44 29
2	mae	0.1726	0.1929	0.1986	0.1693	0.1743	1 16 43 39 7
	rmse	0.2138	0.2388	0.2456	0.2078	0.2121	7 7 11 29 45
3	mae	0.1727	0.1931	0.1993	0.1694	0.1744	7 7 30 7 29
	rmse	0.2142	0.2391	0.2458	0.2085	0.2123	43 16 29 7 7
1	mae	0.1822	0.1813	0.2123	0.1851	0.1766	0 6 0 18 14
	rmse	0.2268	0.2242	0.2598	0.2249	0.2161	0 40 30 0 41
2	mae	0.1822	0.1826	0.2127	0.1856	0.1772	34 12 14 0 41
	rmse	0.2276	0.2255	0.2604	0.2264	0.2166	7 43 14 32 14
3	mae	0.1822	0.1827	0.2129	0.1873	0.1774	14 40 30 23 23
	rmse	0.228	0.2255	0.2609	0.2271	0.2168	14 6 6 18 13
1	mae	0.1905	0.2147	0.1895	0.196	0.2022	0 6 0 18 14
	rmse	0.2318	0.2615	0.2327	0.2345	0.243	0 40 30 0 41
2	mae	0.1935	0.2155	0.1897	0.1974	0.2037	34 12 14 0 41
	rmse	0.235	0.2619	0.234	0.2358	0.2445	7 43 14 32 14
3	mae	0.1945	0.2165	0.1898	0.1983	0.2038	14 40 30 23 23
	rmse	0.2364	0.262	0.2342	0.2379	0.2459	14 6 6 18 13

Table 4.1: Top 3 RNN Models

Conclusion The objective of the research question is to find a model that performs well on its own stations, but also on several stations, in different locations, not so far from the base station. Given enough resources, the parallel results allow us to determine a specific model, best suited for each station. However, a more efficient way would be to find a generalized model achieving as good results across all stations as the results obtained in the parallel performances, hence the cross site results. Looking at the best average of both metrics (**mae** table 4.2 and **rmse** table 4.2) over the five stations, the six following models were isolated (the colored row correspond to the color station the model was trained on) :

Based on the tables above, the best model in terms of **MAE** is the model 7 and in terms of **RMSE**, it is model 44. They have, respectively, one layer with 27 neurons and 2 layers with 7 and 15 neurons. Like in a championship, these intrinsic characteristics made the cut for the next level, which consists of training the model with a smaller batch size and see if they perform better.

The previously mentioned datasets were also experimented during this process. None of the other transformed dataset were able to beat the performance of the

Model	Possession	St Leu	St Andre	Moufia	St Pierre	Avg.
7	0.1727	0.1931	0.1998	0.1694	0.1743	0.1819
0	0.1827	0.1813	0.2123	0.1856	0.1774	0.1879
37	0.1902	0.2147	0.1935	0.196	0.2037	0.1996

Table 4.2: Best MAE for RNN models

Model	Possession	St Leu	St Andre	Moufia	St Pierre	Avg.
44	0.213	0.2401	0.2451	0.2077	0.2143	0.224
0	0.2289	0.2242	0.2617	0.2249	0.2179	0.2315
38	0.2356	0.262	0.235	0.2345	0.2468	0.2427

Table 4.3: Best RMSE for RNN models

”monthly-hourly” Min-Max normalized dataset with ten variables. The errors were worse by 0.01 to 0.05, which in our case, represents a difference.

The final structure of the model will thus have between one and two layers with neurons ranging from 5 to 30. It shows that a smaller number of neurons per layer is better than a larger one. We can consider that each input sample is characterized by only a few features. If we allow the network to remember too many of these features (like increasing the number of neurons higher than 50 neurons), the model will not use all of them and could even be confused by this profusion of information in this situation. This is why this range of small number of neurons will be considered for further experiments. This will also be the starting point of the parameters’ sets for the spatio improved models later on in this chapter.

4.2.2 Convolutional Neural Networks

As for the Recurrent Neural Networks, the Convolutional Neural Networks have a lot of possible parameters for each layer and each network can be composed by an infinite number of layers. To be consistent with the previous learning section, the batch size value is also fixed to a quantity of 64. Thereby, the training time of each model has been reduced drastically, leaving comparison time for several type of networks’ structures.

Parallel Site Referring to the recurrent models, the CNN models seem to behave quite like them. The performance scale for the parallel results is the same as the recurrent ranking. More surprisingly, despite the lack of memory capabilities of the CNN structures, the results in Table 4.4 are really close to the RNN results in Table 4.1. It indicates that, even in time-series sequences, Convolutional Neural Networks models are capable of efficient patterns detection, like in 2D image or 1D sequence

classification. Moreover, this is possible cause the inner structure of the data sequence is appropriate. Indeed, meteorological data features important events and conditions that can be (easily) detected by the convolutional layers of the network. Once again, the last column of the Table 4.4 contains the numbers identifying the best performing models for each station. The reader can find a complete description of each model in the appendix B.

Cross Site Inspecting the cross-station results, they are consistent with the analysis done in the recurrent networks section. The results are ranked by geographical closeness and wind orientation category. For instance, the model trained on Saint-Leu’s data performs better on Saint-Pierre than on Possession and finally on the windward stations, Moufia and Saint-Andre, which is categorically logical. For the St-Andre’s model, unlike the recurrent model, it performs better on its own data rather than on the other ones. Over-fitting can also be one of the consequences for which the cross-site results are better than the parallel results.

		Possession	Saint Leu	Saintt Andre	Moufia	Saint Pierre	Models
1	mae	0.1722	0.1929	0.2001	0.1698	0.1702	22 39 17 67 16
	rmse	0.212	0.2358	0.2433	0.2081	0.2067	67 39 67 17 21
2	mae	0.1731	0.1939	0.2006	0.1703	0.1706	67 26 67 17 21
	rmse	0.2124	0.2366	0.2454	0.2085	0.2075	17 51 17 51 16
3	mae	0.1733	0.1947	0.2008	0.1706	0.174	17 50 22 56 7
	rmse	0.213	0.237	0.2457	0.2088	0.212	22 48 29 67 7
1	mae	0.1824	0.1811	0.2142	0.1873	0.1775	21 27 35 21 7
	rmse	0.226	0.222	0.2588	0.2258	0.2157	27 42 3 7 42
2	mae	0.1827	0.1823	0.2159	0.1882	0.1778	27 7 3 7 3
	rmse	0.2276	0.2221	0.2589	0.2265	0.216	21 27 42 21 3
3	mae	0.1845	0.1828	0.2161	0.1904	0.1798	12 21 34 29 21
	rmse	0.2278	0.2222	0.259	0.2276	0.2166	42 7 35 3 7
1	mae	0.1961	0.2201	0.1914	0.1926	0.2015	21 27 35 21 7
	rmse	0.2376	0.2658	0.2333	0.2322	0.2416	27 42 3 7 42
2	mae	0.1963	0.2206	0.1914	0.1941	0.2042	27 7 3 7 3
	rmse	0.2389	0.266	0.2338	0.2334	0.2428	21 27 42 21 3
3	mae	0.1966	0.2222	0.1918	0.196	0.2047	12 21 34 29 21
	rmse	0.2389	0.2675	0.234	0.2337	0.2464	42 7 35 3 7

Table 4.4: Top 3 CNN Models

Model	Possession	St Leu	St Andre	Moufia	St Pierre	Avg.
67	0.1731	0.198	0.2006	0.1698	0.1746	0.1832
7	0.1823	0.1854	0.2174	0.1882	0.1775	0.1902
6	0.1966	0.2201	0.1948	0.1941	0.2015	0.2014

Table 4.5: Best MAE Models CNN

Conclusion Once again, the best average models were selected out of the mass for the purpose of best structure identification. For each trained station, the results are described in the tables [4.2](#) and [4.3](#). The average performances for the best CNN models are practically identical to the best RNN ones, which show the impressive capabilities of pattern detection of CNN compared to the memory abilities of the RNN. Lets now isolate the best models' structures for the present experiments so to shorten the final prediction trials, they will be done with a smaller batch size. Again, the current dataset was the best performing one among the formerly cited ones.

Model	Possession	St Leu	St Andre	Moufia	St Pierre	Avg.
63	0.212	0.2406	0.2433	0.2088	0.2126	0.2235
3	0.2231	0.2283	0.2588	0.2276	0.216	0.2308
6	0.2405	0.2675	0.2384	0.2337	0.2416	0.2443

Table 4.6: Best RMSE Models CNN

Over all, only four different models detach for the others : models 67, 7, 3 and 6. They all share a similar layer's layout : two consecutive sequences of the layers "Convolution - Convolution - MaxPooling" ended by a flatten layer and a fully connected layer. The first filter parameter ranges from 5 to 30 while the second one spans from 5 to 15. The kernel and pooling sizes are low odd number varying from 3 to 7. The best structures go against one of the general rule of thumb in the CNN building process that is : the deeper the layer, the larger the number of filters. But concur with the rule of thumb stating the use of low odd kernel size parameters. Comparing the best averaged models with the top 3 ranked ones form Table [4.4](#), models 67, 7 and 3, logically, appear many times while model 6 is less present. Nonetheless, the final convolutional neural networks' structures have been identified for the last part of the modelling process.

A thorough comparison between the best **RNN** and **CNN** results is done in the section [4.3.3](#), once the models have been processed with a smaller batch size.

4.3 Improved methods

As we have observed during the results of both types of models, recurrent and convolutional neural networks, some of them perform better on different station's data than the ones they learned from. In a similar way, some convolutional NN performed better than some recurrent NN, while they were learned also from the same data. To profit from these observations, two new type of models will be investigated during this section : a spatio-temporal neural network model, trained on multiple stations in parallel, and a convolutional recurrent neural network using the two types of deep learning layers.

4.3.1 Spatio-Temporal Combinations

It previously occurred for Recurrent and Convolutional models that they performed better on data from other stations. Based on this observation, using Keras, it is possible to combined (some) models trained on different stations into a single new model, see Figure 4.2. With the spatio model, one can decide to predict the target variable of specific stations or all of them (like a sequence). More over, after the fusion of the single models, one can also choose to add new layers, whether it is LSTM or Dense ones. However, these two architectural choices will complexify the overall structure of the model and increase the number of weights to learn, hence, the training time, that is already consequent.

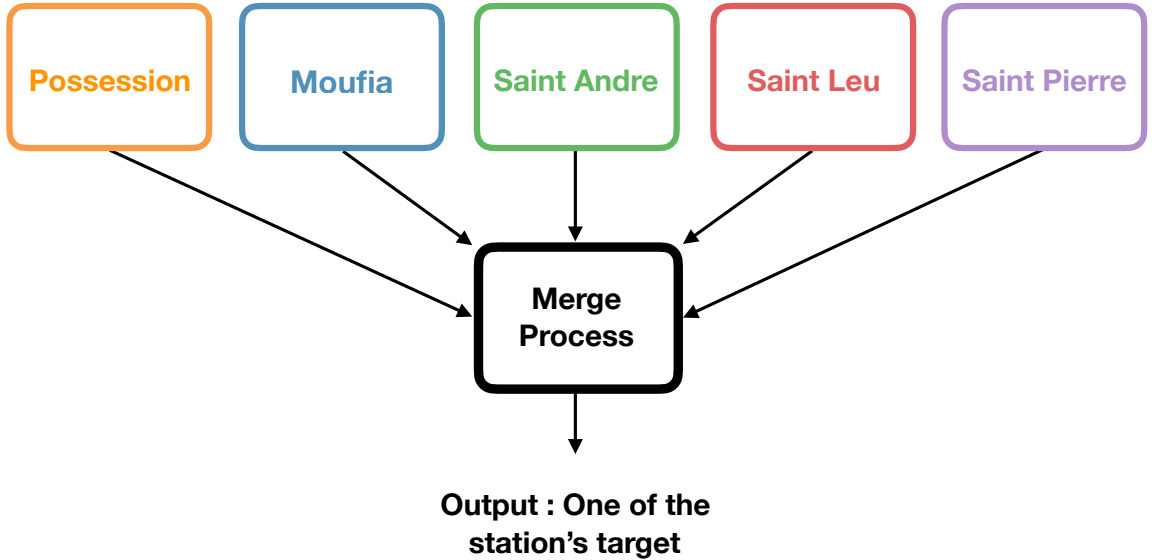


Figure 4.2: Spatio-temporal Model's Structure

To speed up the architecture-training phase, the number of parameters' sets was reduced. The final grid of parameters to validate contains only the best performing sets from each type of models previously tested. Eventually, five and six different sets were defined for the RNN and CNN models, they are respectively presented in Table 4.7 and in Table 4.8. As up to four different model's structures can be experimented

(RNN, RNN-lstm, CNN, CNN-fully³), two stations only were kept to do the different trials: *Possession* and *Saint-Leu*.

RNN

For this experimental part, four different types of structures were tested (listed in the first row of Table 4.7). They consist of a different combination of layers and/or station data taken into consideration. Within each structure, five different sets of hyper-parameters have been analyzed (see in the second row of the table below). To keep a consistent analysis with the previous results, each model has a learning phase of 10 epochs with a batch-size equals to 64, still for computational and time constrains.

	Model 1	Model 2	Model 3	Model 4																		
Structure	All Stations	All Stations + Fully Connected layer	Only 3 stations	3 stations + Fully Connected layer																		
Hyper-parameters	<table><tr><th>Set ID</th><th>LSTM 1</th><th>LSTM 2</th></tr><tr><td>0</td><td>5</td><td>15</td></tr><tr><td>1</td><td>10</td><td>30</td></tr><tr><td>2</td><td>15</td><td>45</td></tr><tr><td>3</td><td>20</td><td>60</td></tr><tr><td>4</td><td>25</td><td>75</td></tr></table>				Set ID	LSTM 1	LSTM 2	0	5	15	1	10	30	2	15	45	3	20	60	4	25	75
	Set ID	LSTM 1	LSTM 2																			
	0	5	15																			
	1	10	30																			
	2	15	45																			
	3	20	60																			
	4	25	75																			
The parameter Set ID indicates the number of set which will be reproduced on the visualization figure below.																						
The parameter LSTM 1 defines the number of neurons the first recurrent layer of the model has. LSTM 2 defines the number of neuron for the second layer.																						

Table 4.7: Sets of RNN parameters

On Figure 4.3, a new type of visualization (somehow following the principle of "small multiples") is displayed. Its aim is to represent a table containing bar-charts with multiple key along the different axis. It exists three types of keys : the station (Saint-Leu, Saint-Pierre, Possession) on the y-axis, the models' structure on the x-axis and different sets of hyper-parameters represented by the bars in the charts. This visualization is experimented to provide a general overlook and a quick analysis of the results.

For the results on Figure 4.3, the models were trained on *Possession*'s data and validated on the three **leeward** stations. In spite of *Moufia*'s results that are not present

³RNN-lstm implies a lstm layer after the merge of the models into the "spatio" model. In the same way, CNN-fully implies a fully connected layer between the merge and the output. In the other case, the output value occurs right after the merge.

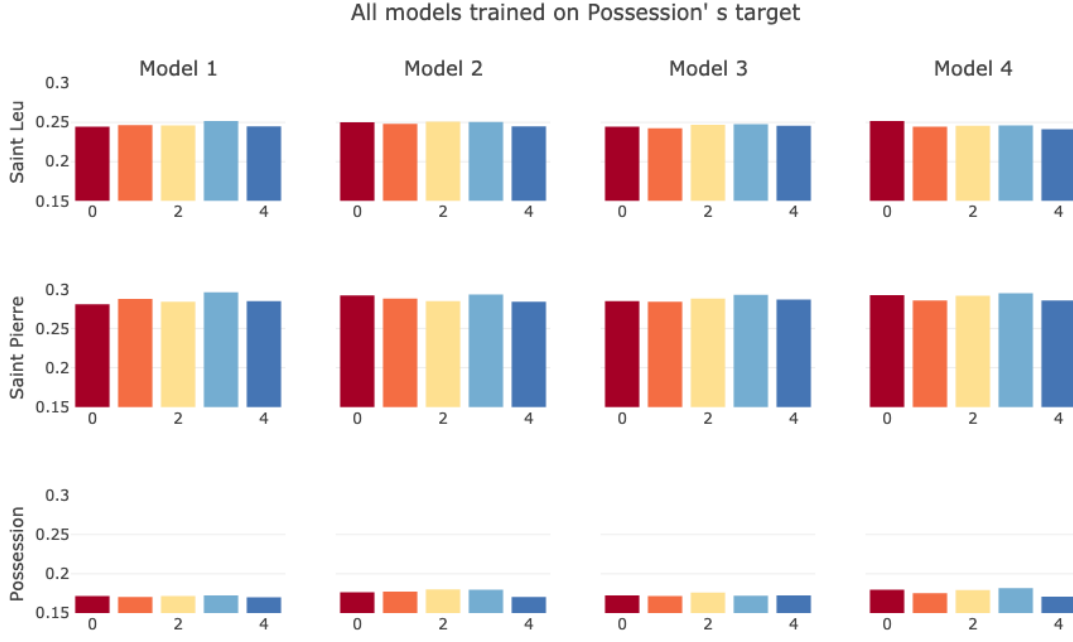


Figure 4.3: Small Multiples **MAE** Results Representation for the Spatio-Recurrent Models on *Possession*. The x-axis of each sub-figure characterizes the **set ID** present in Table 4.7.

here, the models perform better when validated in the continuity of their training. Compared to the **cross** results of the "simple" RNN's tested in Section 4.2.1, the performances of *Possession*'s models on *Saint-Leu* and *Saint-Pierre* are worse. The average **MAE** is around 0.25 for *Saint-Leu* while it is around 0.28 for *Saint-Pierre*. On table 4.1, the best **MAE** results were around 0.192 and 0.1737 respectively for both stations! It clearly indicates that the spatio models do not work efficiently from a **cross** performance perspective. What are the reasons of such deviation between the 2 types of models ? One reason could be that the learning process of the target variable has a completely different effect on the weights of the "spatio" models, due to the additional samples, the models do not cope really well when they are asked to evaluate the target variables coming from other stations.

Looking more specifically at the results from *Possession*'s validation set, we can observe that the differences between the models are all really close, like it is the case for the previous results also. The models with the additional fully connected layer after the "merging" section, achieve barely worse results than the RNN model type 1 and 3. It indicates that this additional layer is not the best structure choice to obtain better performance. However, the models without this new layer show good performances, are over-performing the results of the "simple" RNN from Table 4.1. Set of parameters numbered 2 of the "spatio" structure model 1 gets a **MAE** value of 0.1710 while the best RNN achieved a value of 0.1722. It is not significant but it is still a slight improvement, for more or less the same learning time! This slight improvement could come, first of all, from the addition of the data from other stations. Indeed, it

can help the model to get more informations and potentially locate in space the station relatively to the others. Although more informations doesn't mean it is more useful for the model, it can confuse it if the data are inconsistent or opposite, this is why the data analysis is crucial to detect such inconsistencies. Additionally, as we have uncovered during the data analysis section, the inter-correlation among the variables from different stations are quite high, which means that the information of one variable station is behaving in a similar way than the same variable from another station. It can have 2 effects : one would be that the model is not performing worse because the data are similar. The other one is that the model is not performing significantly better for the same reason that the data are comparable, reaching a paradox situation.

Sadly, these "spatio" models only seem to obtain good predictions when validated on the same station as the one they were trained on, which is not a advantage for generalization. Indeed, the other bar-charts, on Figure 4.3, all indicate larger errors than their respective results from the Recurrent Neural Network Section 4.2.1.

CNN

As for the previous section, the same model's structure have been investigated with six different sets of parameters on two different stations (see Table 4.8 for the exact definition of the different sets of parameter). Indeed, the CNN models train twice faster than the RNN ones, which allow more flexibility for the experimentation part during the allocated time.

	Model 1	Model 2	Model 3	Model 4																																			
Structure	All Stations	All Stations + Fully Connected layer	Only 3 stations	3 stations + Fully Connected layer																																			
Hyper-parameters	<table><tr><th>Set ID</th><th>Filter 1</th><th>Filter 2</th><th>Kernel</th><th>Pool size</th></tr><tr><td>0</td><td>15</td><td>5</td><td>3</td><td>3</td></tr><tr><td>1</td><td>15</td><td>5</td><td>5</td><td>5</td></tr><tr><td>2</td><td>21</td><td>7</td><td>3</td><td>3</td></tr><tr><td>3</td><td>21</td><td>7</td><td>5</td><td>5</td></tr><tr><td>4</td><td>45</td><td>15</td><td>3</td><td>3</td></tr><tr><td>5</td><td>45</td><td>15</td><td>5</td><td>5</td></tr></table>				Set ID	Filter 1	Filter 2	Kernel	Pool size	0	15	5	3	3	1	15	5	5	5	2	21	7	3	3	3	21	7	5	5	4	45	15	3	3	5	45	15	5	5
	Set ID	Filter 1	Filter 2	Kernel	Pool size																																		
	0	15	5	3	3																																		
	1	15	5	5	5																																		
	2	21	7	3	3																																		
	3	21	7	5	5																																		
	4	45	15	3	3																																		
	5	45	15	5	5																																		
The parameter Set ID has the same purpose as for the previous table. The parameters Filter 1 and Filter 2 respectively account for the number of filters in the first and second consecutive layer of the network. Kernel and Pool size have the same meaning as in section 4.2.2 .																																							

Table 4.8: Sets of CNN parameters

The same visualization principle has been adopted for simplicity. In this particular case, thank to the relative efficiency of CNN’s learning, the optimization process has been done on two station’s target sequences : *Possession*, see Figure 4.5 and *Saint-Leu*, see Figure 4.4. For *Saint-Leu*’s models, clearly the models obtained the best results on the former station, with an average best achieved by the model’s structure 4, trained on all the **leeward** stations with a fully connected layer after the merging part of the model. When validated on the 2 other stations, the models performed worse than the non-spatio models, like it is the case for the rnn spatio models. It seems that the combination of multiple data sources and the optimization process on another station’s target than the one tested is also too confusing for the models. Compared over all three stations, the best model structure is the fourth one but the best set of parameters is harder to isolate as all the results, within a model layout, are extremely close to each others. The set 2 is the parallel-best on model 4 but not on the cross performance. However, as any of the cross performances are better than the ones of the non-spatio models, the set of parameters [21, 7, 3, 3] is the best one. It confirms the assumption that not a lot of neurons are needed for the learning process. The models does not need a lot of filters to detect the high level representations of the data.

For *Possession*’s models, on Figure 4.5, the best results also occur on the parallel results thanks to the model trained on all stations with a fully connected layer, the same structure as for *Saint-Leu*’s model. However, looking closely at the different structures, the deviations between the results are once again very difficult to spot ! The results of model’s structure 4 are almost identical to the ones of model’s structure 2, confirming that the additional fully connected layer has a positive impact on the convolutional network, contrarily to the negative impact that it had on the spatio recurrent network. The **MAE** result for this spatio CNN model is **0.1705** while the best ”simple” CNN model achieved a result of **0.1722** on the same metric. Once more, the spatio model seems more efficient than the normal model, thought not by a significant margin. Seeing that the deviations are so limited between all the different sets of parameter, the set of parameters ID 2 is a good compromise between the number of neurons and the training complexity of the models. Although the structure seems to be the best one, one last doubt remains on the number of stations to include in the spatio model. As *Possession* is a the limit of the leeward and windward frontier, it seems the model prefers input data from all the others stations. Here the **wind** category seems less important than the exact location of the *Possession* that is perfectly positioned, on the coast line, between the four others stations.

For a leeward station, the logical mindset would be to only consider the combination of all the leeward stations. In this specific case of the structure 2 performing better than the other ones, we could ask ourselves what would happen if an important meteorological event would occur and would impact the eastern coast but not the western coast ? Would the model cope with such data ? The logical assumption would be to only consider the station from a same wind category or space area to prevent any undesired behaviour.

Overall, the cross results seem to be better when the models were trained on *Saint-Leu* than on *Possession*. It indicates clearly that there is a relationship between the performance of the models and the geographical position of the trained station. In that case,

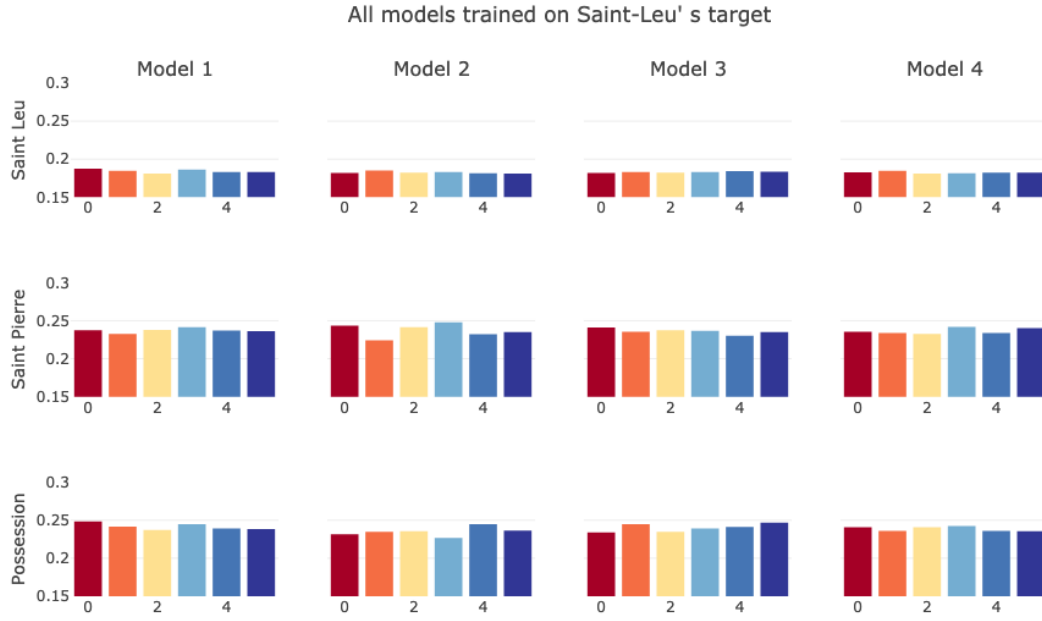


Figure 4.4: Small Multiples **MAE** Results Representation for the Spatio-Convolutional Models on *Saint-Leu*. The x-axis of each sub-figure characterizes the **set ID** present in Table 4.8.



Figure 4.5: Small Multiples **MAE** Results Representation for the Spatio-Convolutional Models on *Possession*. The x-axis of each sub-figure characterizes the **set ID** present in Table 4.8.

Saint-Leu is in between the 2 other tested stations while *Possession* is on the extreme side compared to *Saint-Pierre*. The solar irradiance consequently differs when the station are that far from each other and it can be one of the root cause of such deviations.

4.3.2 CRNN

The Convolutional Recurrent Neural Networks model should capitalize on the advantages of the two previous types of Deep Learning layers, the convolutional and the recurrent ones. On one hand, the convolutional layer will perform an advanced feature extraction of the data, which will serve as the input of the next layer. On the other hand, the next layer will be a recurrent layer that will now operated a recursion step on a more "refined" type of data. Below is a table, briefly, summarizing the best results, once again, obtained on a parallel validation process of three stations.

	Possession	Saint-Leu	Saint-Andre
MAE	0.1738	0.1839	0.1915
RMSE	0.2136	0.2286	0.2355

Table 4.9: Best results for CRNN models

Obviously, the results are promising but not as good as the previous performances obtained with the recurrent neural networks, remaining the best ones. The hopeful assumption of joining the two types of layers could be erroneous or the networks have simple not been fine tuned enough! These CRNN structures, achieving the former results, all have two consecutive pairs of "convolutional-pooling" layers with relatively few neurons (21 for the first layer and 9 for the second one) with a kernel and pooling size of 3. The number of neurons for the two recurrent layers are equivalent to the values of the best performing spatio one and are equal to 10 and 30 neurons for the first and second layers, respectively.

4.3.3 Discussions

During the training and validation processes, a batch size fixed at 64 was used throughout all the models. Since the best models have been selected out for testing, each model will be finally trained, with a batch size equals to 8, on the dataset composed of the training and validation sets and then, its prediction capabilities will be tested on the year 2017, left aside until now.

On Table [4.10](#), the best recurrent and convolutional networks are compared against the best results obtained by *Bruneau and al.* [\[23\]](#), which constitute the baseline for the present work. All models predict the solar irradiance ratio one hour after the most recent available observation and they all base their predictions on one hour worth of observations.

As expected, the deep learning neural networks outperform the two baseline, depicting other machine learning algorithms. XGB model is a gradient boosted decision

	Possession		Saint-Leu		Saint-Andre	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Arima MLP	0.214	0.283	0.243	0.315	0.255	0.332
Arima XGB	0.195	0.245	0.203	0.255	0.208	0.257
CNN	0.1728	0.1932	0.1876	0.2112	0.1939	0.2202
RNN	0.1705	0.1914	0.1826	0.2049	0.1896	0.2163

Table 4.10: Final Results & Comparison

tree algorithm while MLP model designate the basic version of neural network, the multi-layer perceptron. With these final prediction, we see that the recurrent neural networks obtained better results than the convolutional neural networks, though there isn't a huge difference. On one hand, the RNN structure was a network with two recurrent layers consisting of 7 and 15 neurons each. On the other hand, the CNN structure is a stack of two components. Each component is a sequence of two convolution layers, 21 and 7 filters each and a kernel equals to 3, followed by a maximum pooling layer of 5 units.

In order to analyze their performances from a different perspective, let's take a look at the predicted sequences once plotted on the time axis. On Figure ??, some part of the predicted sequences, zoomed on some particular days, are shown. On each sub-figure, the **blue** line represented the true solar irradiance ratio, the **orange** line depicts the sequence predicted by the best recurrent neural network and the **green** line defines the sequence forecast by the best convolutional neural network.

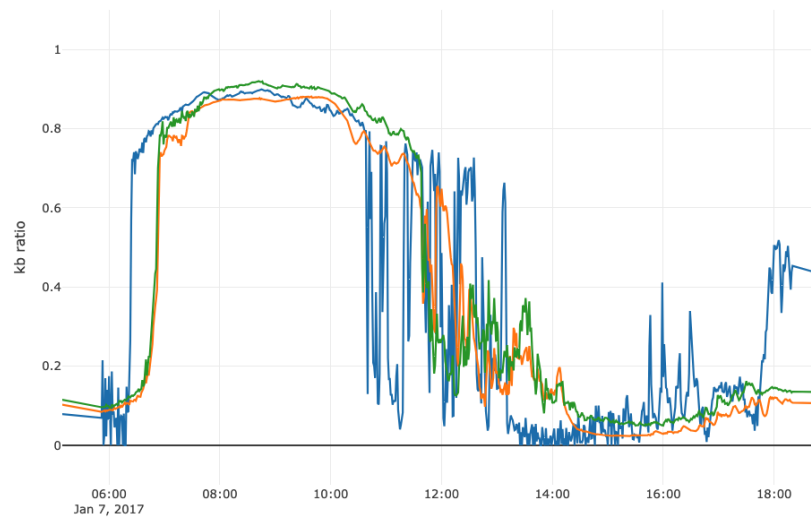
At the beginning of the test set, on Figure 4.6a describing 7th of January, the CNN predictions are a bit greater than the true ones while the RNN predictions are a bit smaller. On the first half of the day, the predictions fit relatively well to the truthful sequence. On the second half, the solar ratio oscillates a bit to tend towards near zero values by the end of the day. Both models understand the general tendency and predict low values of ratio, despite some spikes of the ratio during the evening. On Figure 4.6b, the models also show that they are sensible to rapid variations on days fluctuating between periods of high and low solar ratios, despite some lack of responsiveness.

After a few months deep into the test set, we can see that the pair of models both cope efficiently with the different weather conditions. On April 12th, Figure 4.6c, the model forecast a good level of radiations at the beginning of the day with a slow decrease throughout the day. The model seem insensible to the strong variations, positive or negative surges, and keep the predictions on a general trend, it is like if they knew what is coming! It is a really good sign for the overall purpose of the research question that is to optimize the control of the grid and the management of the electricity production. On Figure 4.6d, corresponding to a cloudy 15th of April, the RNN handles better the behavior of the ratio while the CNN sometimes sparks at undesired

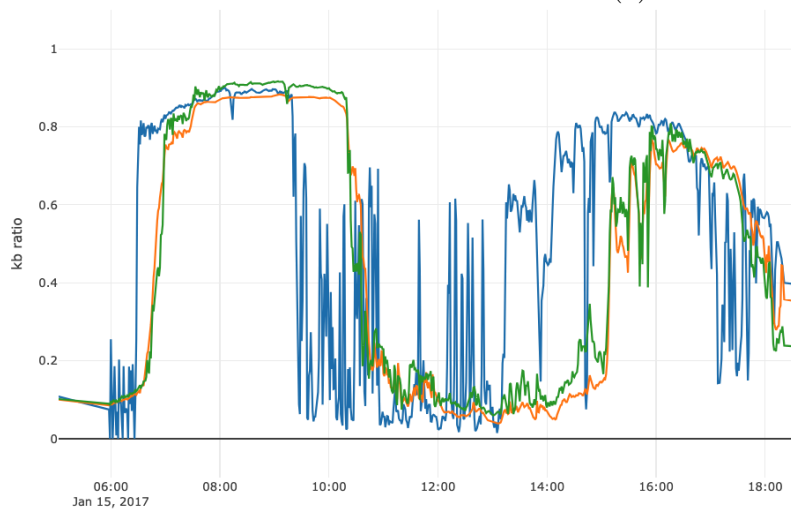
moments. In this specific case, we can observe one downside coming from the discontinuity of the input samples, i.e. remember the night-time values have been removed from the dataset. At the beginning of the day, the solar ratio fluctuates from 0 to 0.5 during 1 hour and an half. Because the model has not a lot of information on the night conditions just before the prediction, it has troubles to correctly adjust its prediction on the actual value. A solution to this issue would be to consider a larger Δp , let's say 120 or 240, than the actual one equal to 60 past observations! However, it comes at an higher computational cost because of the increasing number of weights to train. . .

Finally, more than nine months after receiving the last sample of training, e.g the training data stopped in December 2016, both models seem to perfectly predict the solar ratio on Figure 4.6e. In December, on Figure 4.6f, despite some inconsistencies, the models are far from being off target and they both draw a global pattern the solar ratio is heading towards.

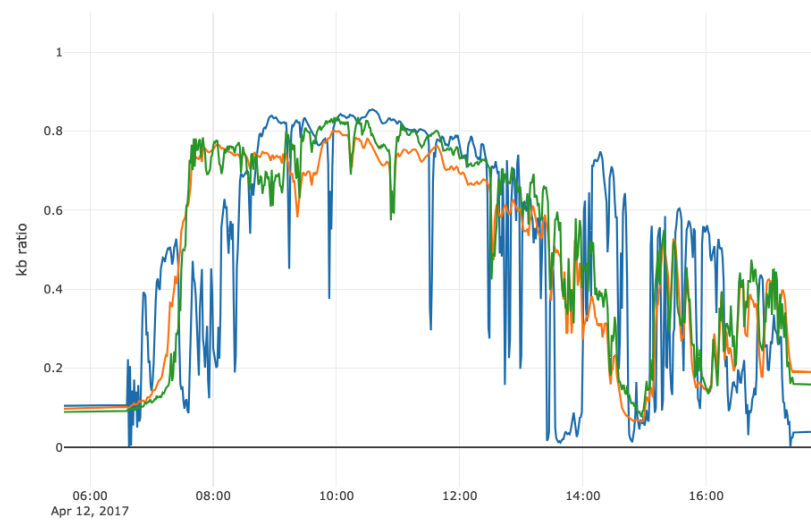
To resume, thanks to its tendency to predict a higher ratio on sunny days, the convolutional neural network is achieving better results on these specific days. However, the recurrent neural network is coping more effectively on days with overcast, when the solar ratio fluctuates at a quicker rate. For some days even, it seems like the average of the two prediction would give a better prediction than taken individually, both values having a regulatory effect on one another.



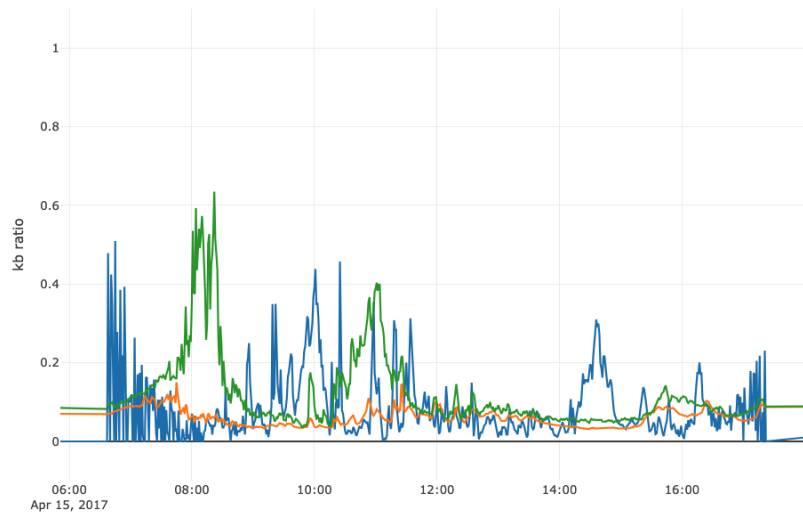
(a) 7th January 2017



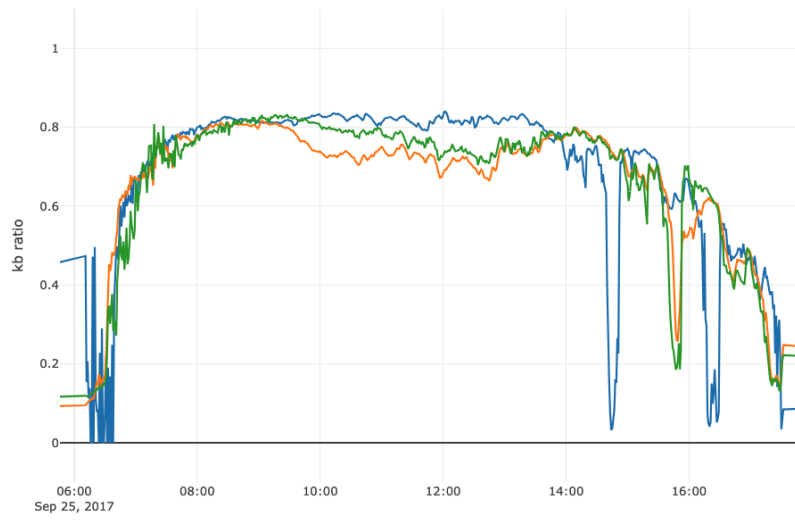
(b) 15th January 2017



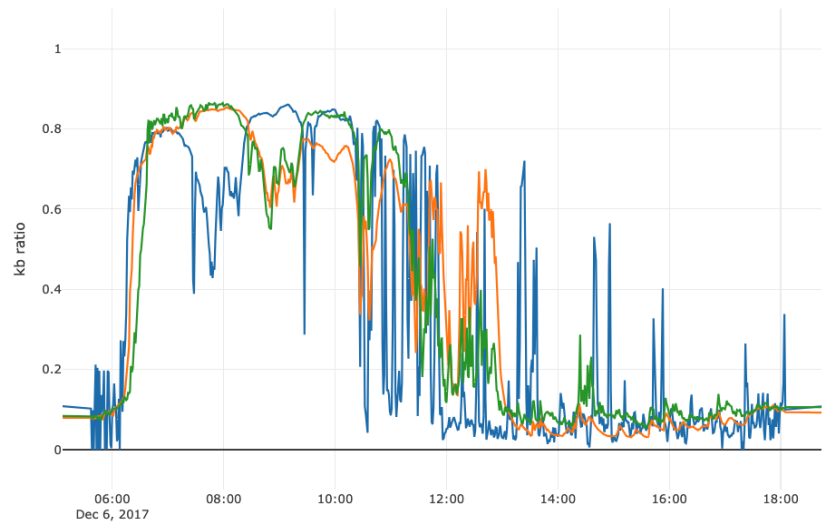
(c) 12th April 2017



(d) 15th April 2017



(e) 25th September 2017



(f) 6th December 2017

Chapter 5

Conclusion

During this research work, we introduced the rising field of Machine Learning with a particular focus on Deep Neural Networks applied to a meteorological challenge. As the predictions of meteorological events are, nowadays more than ever, at the core of every society around the world, we applied Deep Learning algorithms on historical weather data to predict a specific meteorological variable, the **solar irradiance**, one hour into the future.

The objectives of the thesis were **twofold**. Firstly, we built deep neural networks such as the recurrent and convolutional neural networks to predict the solar irradiance ratio with an horizon of one hour. This operation is based on the last hour of available data and is due to optimize the utilization and production of renewable energy generated by photo-voltaic grid. These techniques were expected to improve, both numerically and computationally, the current solutions such as Numerical Weather Simulations, satellite imagery analysis and traditional Machine Learning algorithms like the Decision Trees or the Multi Layer Perceptron neural networks. Secondly, holding datasets from 5 different locations on *La Reunion* island, we wanted to find clever ways to improve the deep learning models combining the temporal and the spatial dimensions of the data.

The models and results obtained with the validation procedure **met our expectations**. We were able to reduce the MAE errors by **3 to 4%** while greatly improving the RMSE results **up to 10%** for some stations, reducing the gap between the predicted and truthful values of solar ratio. Without any particular transformations of the data, the Recurrent Neural Network was able to concretely replicate the human brain capabilities of ***Recall*** and ***Recognition***, demonstrated by the adaptability to react to the changing meteorological conditions. Thanks to massive amounts of historical data, the model is able to detect and remember the important features from past events and to reflect the behavior of future events to correctly forecast the solar irradiance with the required intensity. The predictions are done at one minute interval as soon as new sample are easily recorded using ground based instruments. The Convolutional Neural Network uses the same data in a different way. There are no feedback connections within the neurons but they are made more efficient using advanced pattern detection techniques enabling features within the time series sequence to be more easily identified and recognized in future observations. The final structures of the two different types of models **outperformed** the baseline results using the same

data, proving the potential such methods possess in the meteorological field.

Furthermore, some improvements like the spatio-temporal combination of the data showed some promising advantages to join the time-series data with the spatial dimension of several weather locations to enhance the prediction of the solar ratio. However, this combination of spatial locations **needs to be carefully operated**. On the specific case of *La Reunion* island, the data were consistent between the stations and could be "blended" together, but the combination could surely be improved, tuning a bit more how theses models are built! In other parts of the world and at larger scale, this operation would require deeper analysis on a case by case basis for the collection of available data, leading in some situations to an incompatibility if the chosen stations are too singular!

Eventually, the same observation can be made for the general structure of the Deep Neural Networks. The final structures of the models in this situation can not be directly translated to other location in the world and used as is. In the present work, we showed that the concept of Deep Neural Networks could also be applied in the meteorological field as it is efficiently applied in the natural language processing field or in the image processing world. However, adjustments would need to be carefully considered for each specific dataset.

Throughout this work, we demonstrated that deep learning neural network could be highly efficient and accurate, within a reasonable amount of training time considering the volume of data and given the appropriate computation equipment, for the prediction of such volatile target variable that is the **solar irradiance ratio**. These methods are consequently convenient solutions to meet the requirements of the technical problems societies, around the world, are facing in matter to achieve more profitable and cleaner usage of the renewable energies.

Perspective and Future Works

The present work is the result of many ideas and concepts learned over the past two years. However, some experiments, despite their high potentials, were drafted but they haven't been fully developed and studied. Here are some future work leads.

In particular, facing this quantity of data, **Online Training** could be a proper solution for a dynamic model. Indeed, as meteorological events are not fixed in time, it would be a good idea to continuously train the models with newly recorded data as the model advances in time. This could lead to more precise predictions thanks to events recently learned. A scale between a week and a month of new data seems to be a good start to consider the repeated training.

Additionally, the meteorological scientific field is vast. The models of the research were trained with up to ten variables. However, given the proper resources, **complementary variables** could be sampled and added to the dataset. For example, the current visibility, the dew point or precipitation accumulation could have a significant impact on the prediction of the irradiance ratio as they could be good indicators on the quality

of the day to come. The Deep Neural Networks can also be augmented by static information relative to the geographical and topological characteristics of the research area.

Finally, the horizon of prediction could be another important parameter of the system. In our specific case, we dealt with an horizon of one hour. However, greater horizons could be studied. Naturally, with different horizons, one could wonder if the performances would increase or decrease. Alongside this technical aspect, the practical one implies a different usage of the prediction. With a prediction of half a day or up to 24 hours, we could use the prediction in a totally different way, planning the production of energy at a larger scale for example.

Personal Achievements

This work is by far the biggest one I had to perform during my studies and it taught me a lot of useful outcomes. Throughout this thesis, I had the possibility to apply many things I learned during my degree. It proved me that a careful planning, an imaginative and critical thinking and a scientific rigor are key components to carry out such work.

We are still living in a world where the algorithms and the data scientist engineer need to work in unison to achieve good results. The algorithms would definitively not perform as well as they could if they were given raw data. The prior human analysis is essential for the completeness of the project. On the same note, the human could not process as many data as the algorithms do. This is why the combination of the two "brains" is such a perfect fit and that two heads are better than one.

On top of that, this work truly shows a decisive quality every data scientist should have, the adaptability to the data domain. As an engineer, I did not follow physics studies to become a meteorologist. Anyway, I was able to extensively analyze the data, which is an important part of the work and in line with the analysis results, I built models to correctly predict the solar irradiance variable. It shows that an intrinsic knowledge of the research field is not a necessary condition to deal with such problems. However, another part of the engineer work is also to get acquainted with the research area. The methodology and models used throughout this work could easily be translated to different activity fields. In financial engineering, predictions of economical and stock market variables are often based on historical data. In the medical domain, the vital data of a patient could be analyzed in real time by Deep Learning algorithms to prevent strokes or any other life-threatening complications. These are only two examples of many more where Recurrent and Convolutional Neural Networks could be successfully applied.

At last, I strongly believe in the capabilities of Deep Learning algorithms associated with large and rich datasets to accomplish future prowess. A sure thing is that many solutions involving Machine Learning techniques will continue to pop up in all the scientific engineering fields, leading the world towards a cleaner and safer environment. There are a lot of chances, the meteorological technologies will take part in it.

Appendices

Appendix A

Activation Functions

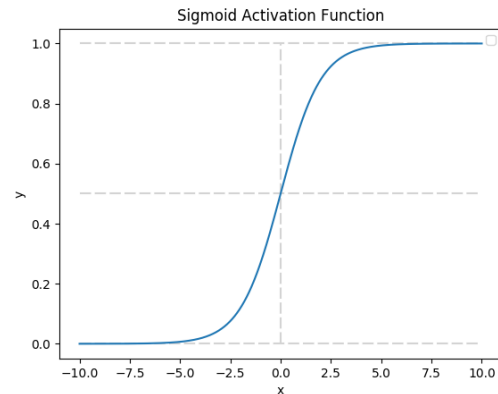
An activation function is used to map the output of the summing function between values such as 0 and 1 or -1 and 1. As mentioned in Chapter 2, many different functions exist to achieve such results, some are presented below. All three functions below are monotonic. They all have a continuous derivative function.

A.1 Sigmoid Function

The equation of the sigmoid is the following one :

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (\text{A.1})$$

As one can see, the output will be mapped between 0 and 1, which make this function useful for ratios or probability prediction.

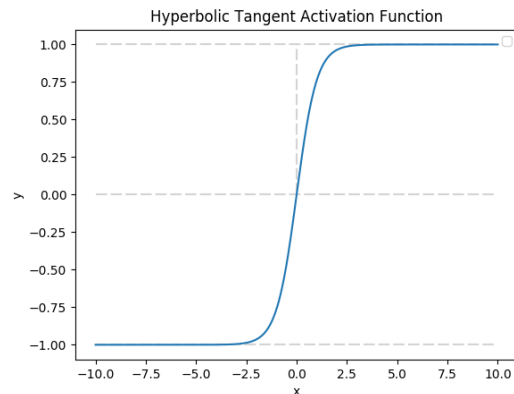


A.2 Hyperbolic Tangent Function

The $\tanh$ function is somehow an improved version of the sigmoid function as it maps the input on a range between -1 and 1.

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \quad (\text{A.2})$$

The improvement comes from the fact that the negative inputs will be mapped to negative output, which was not the case for the *sigmoid* function.

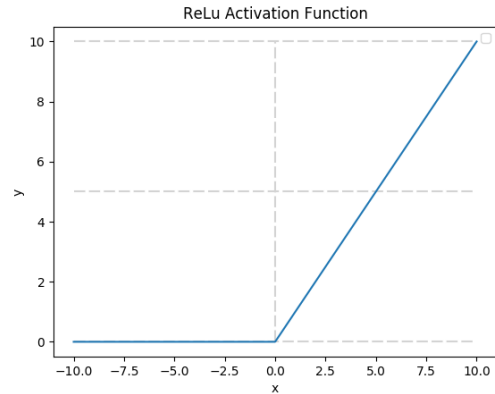


A.3 Rectified Linear Unit Function

The ReLu function is another widely used activation function and its behaviour is ruled by the subsequent equation.

$$ReLU(x) = \max(0, x) \quad (\text{A.3})$$

Unlike the 2 previous functions, the function is not "S-shaped". It maps any negative values to 0 and keep the positive values as is. It is more of the time used in **CNN**.



Appendix B

Models Structure

On the tables below, one can find the structure of the models mentioned in Table 4.1 and 4.4. The different letters correspond to the following layer types:

- L_x : Long Short Term Memory layer where x corresponds to the number of neurons.
- O_x : Dropout layer randomly setting a fraction rate of $x\%$ of input units to 0 at each update during training time, which helps prevent over-fitting.
- D_x : Fully Connected layer where x defines the number of neurons in the layer.
- $C_{x,y}$: Convolutional layer where x is the number of filters and y is the kernel size of each filter.
- P_x : Maximum Pooling layer where x is the pooling size.
- F_x : Flatten layer without any hyper-parameters.

B.1 Recurrent Neural Networks

Model ID	Model Structure	Model ID	Model Structure
0	L300_O20_L600_O20_D1	23	L150_O20_L300_O20_D1
1	L55_O20_L27_O20_D1	24	L225_O20_L450_O20_D10_D1
2	L95_O20_L47_O20_D10_D1	25	L600_O20_L300_O20_D10_D1
3	L75_O20_L150_O20_D1	26	L55_O20_L27_O20_D10_D1
4	L750_O20_L375_O20_D10_D1	27	L750_O20_L375_O20_D1
5	L75_O20_L150_O20_D10_D1	28	L27_O20_L55_O20_D10_D1
6	L75_O20_D1	29	L55_O20_D1
7	L27_O20_D1	30	L47_O20_D1
8	L300_O20_L150_O20_D1	31	L95_O20_D1
9	L375_O20_L750_O20_D1	32	L15_O20_D1
10	L600_O20_L300_O20_D1	33	L150_O20_L300_O20_D10_D1
11	L7_O20_L15_O20_D10_D1	34	L450_O20_L225_O20_D10_D1
12	L225_O20_L450_O20_D1	35	L375_O20_L750_O20_D10_D1
13	L300_O20_D1	36	L375_O20_D1
14	L225_O20_D1	37	L47_O20_L95_O20_D1
15	L15_O20_L7_O20_D10_D1	38	L7_O20_D1
16	L47_O20_L95_O20_D10_D1	39	L750_O20_D1
17	L150_O20_L75_O20_D10_D1	40	L95_O20_L47_O20_D1
18	L450_O20_L225_O20_D1	41	L450_O20_D1
19	L300_O20_L150_O20_D10_D1	42	L150_O20_D1
20	L600_O20_D1	43	L15_O20_L7_O20_D1
21	L150_O20_L75_O20_D1	44	L7_O20_L15_O20_D1
22	L300_O20_L600_O20_D10_D1	45	L27_O20_L55_O20_D1

Table B.1: Structure of the RNN models

B.2 Convolutional Neural Networks

Model ID	Model Structure
0	C_91_7_C_45_7_P_5_C_91_7_C_45_7_P_5_F_D_1
1	C_183_7_C_93_7_P_3_C_183_7_C_93_7_P_3_F_D_1
2	C_15_5_C_9_5_P_3_C_15_5_C_9_5_P_3_F_D_1
3	C_45_3_C_15_3_P_5_F_D_1
4	C_3_3_C_5_3_P_3_C_3_3_C_5_3_P_3_F_D_1
5	C_93_5_P_3_C_93_5_P_3_F_D_1
6	C_3_5_C_5_5_P_3_C_3_5_C_5_5_P_3_F_D_1
7	C_15_5_C_9_5_P_3_F_D_1
8	C_5_3_C_3_3_P_3_F_D_1
9	C_93_5_C_45_5_P_3_C_93_5_C_45_5_P_3_F_D_1
10	C_7_3_C_3_3_P_5_F_D_1
11	C_31_5_C_15_5_P_3_C_31_5_C_15_5_P_3_F_D_1
12	C_31_5_P_3_C_31_5_P_3_F_D_1
13	C_61_7_P_3_C_61_7_P_3_F_D_1
14	C_31_7_P_3_C_31_7_P_3_F_D_1
15	C_15_3_P_3_C_15_3_P_3_F_D_1
16	C_21_3_C_9_3_P_5_C_21_3_C_9_3_P_5_F_D_1
17	C_15_5_P_3_C_15_5_P_3_F_D_1
18	C_15_5_C_31_5_P_3_C_15_5_C_31_5_P_3_F_D_1
19	C_61_7_C_31_7_P_3_C_61_7_C_31_7_P_3_F_D_1
20	C_279_7_P_5_C_279_7_P_5_F_D_1
21	C_15_3_C_5_3_P_5_C_15_3_C_5_3_P_5_F_D_1
22	C_15_3_C_7_3_P_5_C_15_3_C_7_3_P_5_F_D_1
23	C_31_7_C_93_7_P_5_C_31_7_C_93_7_P_5_F_D_1
24	C_5_3_C_15_3_P_5_C_5_3_C_15_3_P_5_F_D_1
25	C_93_7_C_31_7_P_5_C_93_7_C_31_7_P_5_F_D_1
26	C_3_5_C_7_5_P_5_C_3_5_C_7_5_P_5_F_D_1
27	C_7_3_C_15_3_P_5_C_7_3_C_15_3_P_5_F_D_1
28	C_5_3_C_3_3_P_3_C_5_3_C_3_3_P_3_F_D_1
29	C_15_3_C_5_3_P_5_F_D_1

Table B.2: Structure of the CNN models : Part 1

Model ID	Model Structure
30	C_3.3_P_5_C_3.3_P_5_F_D_1
31	C_45.3_C_21.3_P_5_F_D_1
32	C_7.3_C_3.3_P_5_C_7.3_C_3.3_P_5_F_D_1
33	C_7.3_P_5_C_7.3_P_5_F_D_1
34	C_21.5_C_9.5_P_5_F_D_1
35	C_3.3_P_3_C_3.3_P_3_F_D_1
36	C_5.5_C_3.5_P_3_C_5.5_C_3.5_P_3_F_D_1
37	C_3.3_C_7.3_P_5_C_3.3_C_7.3_P_5_F_D_1
38	C_7.5_C_3.5_P_5_C_7.5_C_3.5_P_5_F_D_1
39	C_15.3_C_9.3_P_3_C_15.3_C_9.3_P_3_F_D_1
40	C_7.5_P_5_C_7.5_P_5_F_D_1
41	C_3.5_P_3_C_3.5_P_3_F_D_1
42	C_31.5_C_15.5_P_3_F_D_1
43	C_3.5_P_5_C_3.5_P_5_F_D_1
44	C_21.5_C_9.5_P_5_C_21.5_C_9.5_P_5_F_D_1
45	C_61.7_C_31.7_P_3_F_D_1
46	C_183.7_C_93.7_P_3_F_D_1
47	C_279.7_C_93.7_P_5_C_279.7_C_93.7_P_5_F_D_1
48	C_15.3_C_7.3_P_5_F_D_1
49	C_45.7_C_91.7_P_5_C_45.7_C_91.7_P_5_F_D_1
50	C_45.3_C_15.3_P_5_C_45.3_C_15.3_P_5_F_D_1
51	C_21.3_C_9.3_P_5_F_D_1
52	C_45.7_P_5_C_45.7_P_5_F_D_1
53	C_31.7_C_61.7_P_3_C_31.7_C_61.7_P_3_F_D_1
54	C_45.3_P_5_C_45.3_P_5_F_D_1
55	C_15.3_P_5_C_15.3_P_5_F_D_1
56	C_31.7_P_5_C_31.7_P_5_F_D_1
57	C_45.3_C_21.3_P_5_C_45.3_C_21.3_P_5_F_D_1
58	C_91.7_P_5_C_91.7_P_5_F_D_1
59	C_93.7_P_5_C_93.7_P_5_F_D_1
60	C_273.7_C_135.7_P_5_C_273.7_C_135.7_P_5_F_D_1

Table B.3: Structure of the CNN models : Part 2

Model ID	Model Structure
60	C_273_7_C_135_7_P_5_C_273_7_C_135_7_P_5_F_D_1
61	C_21_3_P_5_C_21_3_P_5_F_D_1
62	C_21_5_P_5_C_21_5_P_5_F_D_1
63	C_91_7_C_45_7_P_5_F_D_1
64	C_5_3_P_3_C_5_3_P_3_F_D_1
65	C_93_5_C_45_5_P_3_F_D_1
66	C_5_3_P_5_C_5_3_P_5_F_D_1
67	C_15_3_C_9_3_P_3_F_D_1
68	C_93_7_C_31_7_P_5_F_D_1
69	C_273_7_P_5_C_273_7_P_5_F_D_1
70	C_7_5_C_3_5_P_5_F_D_1
71	C_5_5_C_3_5_P_3_F_D_1
72	C_5_5_P_3_C_5_5_P_3_F_D_1
73	C_183_7_P_3_C_183_7_P_3_F_D_1

Table B.4: Structure of the CNN models : Part 3

Bibliography

- [1] John A. Lee. LDATE 2010 Information Visualization. pages 26–28, 2018-2019 Q1.
- [2] Simon S. Haykin. Neural networks and learning machines. 2009.
- [3] Lisa Alter. Meteorology. Available online: <http://teachersinstitute.yale.edu/curriculum/units/1994/5/94.05.01.x.html>, 1994.
- [4] H Howard Frisinger. *History of Meteorology to 1800*. Springer, 2018.
- [5] Bernard Marr. 27 incredible examples of ai and machine learning in practice. Available online: <https://www.forbes.com/sites/bernardmarr/2018/04/30/27-incredible-examples-of-ai-and-machine-learning-in-practice/#4010ef107502>, 2018.
- [6] Luxembourg Institute of Science and Technology. Environmental informatics. Available online: <https://www.list.lu/en/research/erin/environmental-informatics/>.
- [7] DK Chaturvedi et al. Solar power forecasting and planning using soft computing approach. 2017.
- [8] Peter Lynch. The origins of computer weather prediction and climate modeling. *Journal of Computational Physics*, 227(7):3431–3444, 2008.
- [9] Richard Perez, Pierre Ineichen, Robert Seals, Joseph Michalsky, and Ronald Stewart. Modeling daylight availability and irradiance components from direct and global irradiance. *Solar energy*, 44(5):271–289, 1990.
- [10] Benoît Molineaux, Pierre Ineichen, and Jean-Jacques Delaunay. Direct luminous efficacy and atmospheric turbidity—improving model performance. *Solar Energy*, 55(2):125–137, 1995.
- [11] J Jensenius and GF Cotton. The development and testing of automated solar energy forecasts based on the model output statistics (mos) technique. In *1st Workshop on terrestrial solar resource forecasting and on use of satellites for terrestrial solar resource assessment*, Washington, DC, 1981.
- [12] RJ Aguiar, Manuel Collares-Pereira, and JP Conde. Simple procedure for generating sequences of daily radiation values using a library of markov transition matrices. *Solar Energy*, 40(3):269–279, 1988.

- [13] TN Goh and KJ Tan. Stochastic modeling and forecasting of solar radiation data. *Solar Energy*, 19:755–757, 1977.
- [14] Solar radiation graphs. *Solar Energy*, 9, 1965.
- [15] RK Swartman and O Ogunlade. Correlation of solar radiation with common parameters in toronto, canada. *Solar Energy*, 13(3):345–347, 1971.
- [16] Annette Hammer, Detlev Heinemann, Elke Lorenz, and Bertram Lücke. Short-term forecasting of solar radiation: a statistical approach using satellite data. *Solar Energy*, 67(1-3):139–150, 1999.
- [17] S Safi, A Zeroual, and M Hassani. Prediction of global daily solar radiation using higher order statistics. *Renewable Energy*, 27(4):647–666, 2002.
- [18] G Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.
- [19] TC McCandless, SE Haupt, and George Spencer Young. A model tree approach to forecasting solar irradiance variability. *Solar Energy*, 120:514–524, 2015.
- [20] Lanre Olatomiwa, Saad Mekhilef, Shahaboddin Shamshirband, Kasra Mohammadi, Dalibor Petković, and Ch Sudheer. A support vector machine–firefly algorithm-based model for global solar radiation prediction. *Solar Energy*, 115:632–644, 2015.
- [21] Shah Alam, SC Kaushik, and SN Garg. Computation of beam solar radiation at normal incidence using artificial neural network. *Renewable Energy*, 31(10):1483–1491, 2006.
- [22] Yashwant Kashyap, Ankit Bansal, and Anil K Sao. Solar radiation forecasting with multiple parameters neural networks. *Renewable and Sustainable Energy Reviews*, 49:825–835, 2015.
- [23] Pierrick Bruneau, Philippe Pinheiro, and Yoann Didry. Data-driven forecasting of solar irradiance. *arXiv preprint arXiv:1801.03373*, 2018.
- [24] Adel Mellit, Soteris A Kalogirou, L Hontoria, and Sulaiman Shaari. Artificial intelligence techniques for sizing photovoltaic systems: A review. *Renewable and Sustainable Energy Reviews*, 13(2):406–419, 2009.
- [25] Thanasis G Barbounis, John B Theocharis, Minas C Alexiadis, and Petros S Dokopoulos. Long-term wind speed and power forecasting using local recurrent neural network models. *IEEE Transactions on Energy Conversion*, 21(1):273–284, 2006.
- [26] Ahmad Alzahrani, Pourya Shamsi, Cihan Dagli, and Mehdi Ferdowsi. Solar irradiance forecasting using deep neural networks. *Procedia Computer Science*, 114:304–313, 2017.
- [27] Yunjie Liu, Evan Racah, Joaquin Correa, Amir Khosrowshahi, David Lavers, Kenneth Kunkel, Michael Wehner, William Collins, et al. Application of deep convolutional neural networks for detecting extreme weather in climate datasets. *arXiv preprint arXiv:1605.01156*, 2016.

- [28] Benjamin Klein, Lior Wolf, and Yehuda Afek. A dynamic convolutional layer for short range weather prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [29] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545, 2014.
- [30] Peter J Brockwell, Richard A Davis, and Matthew V Calder. *Introduction to time series and forecasting*, volume 2. Springer, 2002.
- [31] National Aeronautics and Space Administration. The sunspot cycle, 20170315.
- [32] CarSalesBase. Us car sales data - tesla.
- [33] Francis J Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1):17–21, 1973.
- [34] Shunichi Amari et al. *The handbook of brain theory and neural networks*. MIT press, 2003.
- [35] Abhishek Sharma. Quoras - what is the differences between artificial neural network (computer science) and biological neural network? Available online: <https://www.quora.com/What-is-the-differences-between-artificial-neural-network-computer-science-and-biological-neural-network>, 19 October 2017.
- [36] David A Drachman. Do we have brain to spare?, 2005.
- [37] Luke Mastin. Human memory : Neurons and Synapses. http://www.human-memory.net/brain_neurons.html, 2019.
- [38] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772, 2014.
- [39] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.
- [40] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.
- [41] Alex Graves. Supervised sequence labelling. In *Supervised sequence labelling with recurrent neural networks*, pages 5–13. Springer, 2012.
- [42] Christopher Olah. Understanding lstm networks. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>, August 2015.
- [43] Luke Mastin. Human memory : Memory Recall and Retrieval. http://www.human-memory.net/processes_recall.html, 2019.

- [44] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [45] FT Exploring Sciences and Technology. Direct, diffuse and reflected radiation.
- [46] Pierrick Bruneau, Laurence Boudet, and Cécilia Damon. Neural architectures for global solar irradiation and air temperature prediction. In *International Conference on Artificial Neural Networks*, pages 548–556. Springer, 2012.
- [47] Urvashi Jaitley. Medium — why data normalization is necessary for machine learning models. Available online: <https://medium.com/@urvashilluniya/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029d>, 2019-05-067.
- [48] François Chollet et al. Keras. <https://keras.io>, 2015.
- [49] Météo France La Réunion. Bilans et records météorologiques sur l’île de la réunion. Available online: <http://www.meteofrance.re//climat/description-du-climat>, 2019.
- [50] Python Core Team (2015). *Python: A dynamic, open source programming language*. Python Software Foundation., 2015.
- [51] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.
- [52] Janet Wesner. Medium — mae and rmse - which metric is better? Available online: <https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d>, 2019-04-17.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/epl