POLITECNICO DI TORINO

Master Course in ICT for Smart Socities

Master Degree Thesis

Machine Learning For Predicting Renewable Energy Generation

Supervisor:
Prof. *Michela Meo*
Assistant Supervisor:
Ph.D. *Greta Vallero*

Candidate:
Maurizio Floridia
Matr. s254283

Academic Year 2018-2019

Contents

Abstract

The increasing need to preserve the environment by reducing the emissions of CO_2 and other greenhouse gases highlighted the need work in the energy field. It embraces many other sectors, such as transport, building, ICT and the need for energy management is going to become preponderant, above all with the penetration of renewable energy resources and with the advent of the Smart Grids. This work studies the problem of predicting renewable energy generation focusing on solar energy. The thesis investigates if physical data, such as the Global Solar Irradiance and its components, allow to obtain good performances in predicting solar energy generation. The solution of the problem has been reached by using two types of machine learning techniques, Feed-Forward and Recurrent Neural Network. With them, several training approaches were tested and the Recurrent NN was the one which performed better. Then, it was tested in different locations and time granularity, hourly and half hour. In the end, Neural Networks results extremely effective in predicting renewable energy generation and the work can positively affect several domains, such as pricing strategies, critical situation, emergencies during a blackout, and in energy management and for cost reduction.

CHAPTER 0

Introduction

IPCC (Intergovernmental Panel on Climate Change) defined climate change as a variation in either the mean state of the climate or its variability for an extended period (decades or longer), and it may be due to whether natural internal processes of the earth or human activities. The idea of anthropogenic changes spread out and recently penetrated popular culture, with a growing political resonance. Indeed, humans are increasingly influencing the climate and the earth's temperature. The global raising temperature is affected by the enormous amount of emission gases produced by human activities, responsible for 64% of man-made global warming. The most popular activities involve cutting down forests, livestock farming, burning fossil fuel, and Figure 1 shows quantitative values of CO_2 emissions due to human activities.

Energy efficiency improvements, technological advancement, market dynamics, worries about climate change are leading during last years to a decoupling between CO_2 emissions and economic growth. They aim to carry on decarbonization or a huge reduction of CO_2 emissions. In parallel, however, technological systems are continuously increasing their environmental impact and from one side it is due to computational improvements which enable the release of new services; on the other side, it is due to the exponential usage from people of these services. For instance, the advent of Internet during the '90s, and consequently of the Cloud Computing, Internet Of Things (IoT), Social Media and so on, increased the access and the

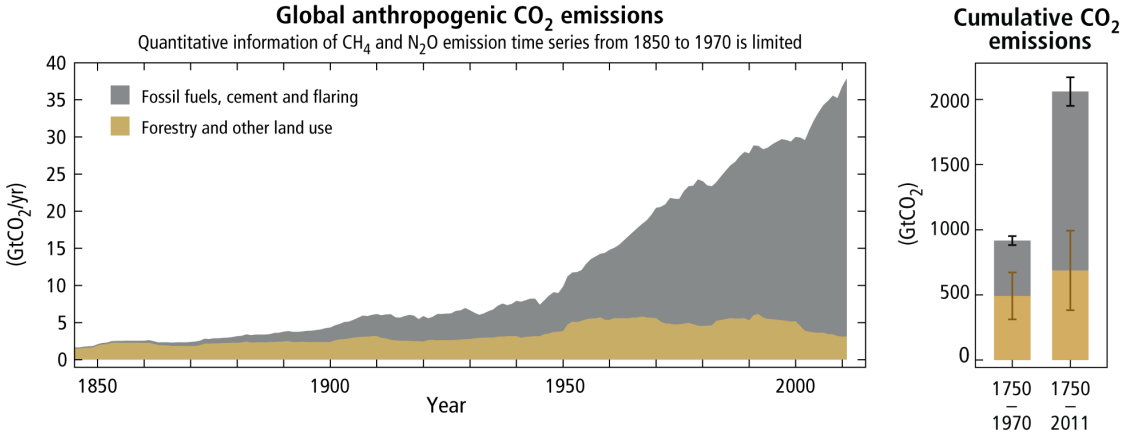


Figure 1: Source: IPCC - Climate Change 2014: Synthesis Report

usage of the Information Technology (IT) infrastructure (Figure 2), rising the environmental damage. Cooperation between energy systems and technological systems is requested, so as policies and strategies at global scales are needed to effectively reduce the environmental impact. In this context, the usage of renewable resources could allow achieving the objective, by making sustainable both technological and energy systems, leading the digital transformation more cleanly. Renewable Energy Sources (RES) can boast to be cleaner, sustainable several fields, with a much lesser negative impact than fossil fuel, and above all, they are infinite [1]. By contrast, they suffer from intrinsic characteristics. For instance, wind and solar power are infinite energy, but their intermittent availability due to the weather dependence discourages the use in some cases, in which the steady presence of energy is highly valuable. However, both sources are easily predictable and this capability is the one which enables several fields in new researches investments for studying many possible applications. With the advent of the smart grid, the concept of "on-grid" and "off-grid" came.
The former is considered whenever many loads take electrical energy from the electrical grid, in which nowadays the energy is mainly produced by fossil fuel, even though the RES is going to introduce in it. The latter takes into account a load whose energy is fed employing renewable energy, which is not linked to the main grid, such as wind or solar energy. The "off-grid" concept is highly considered in

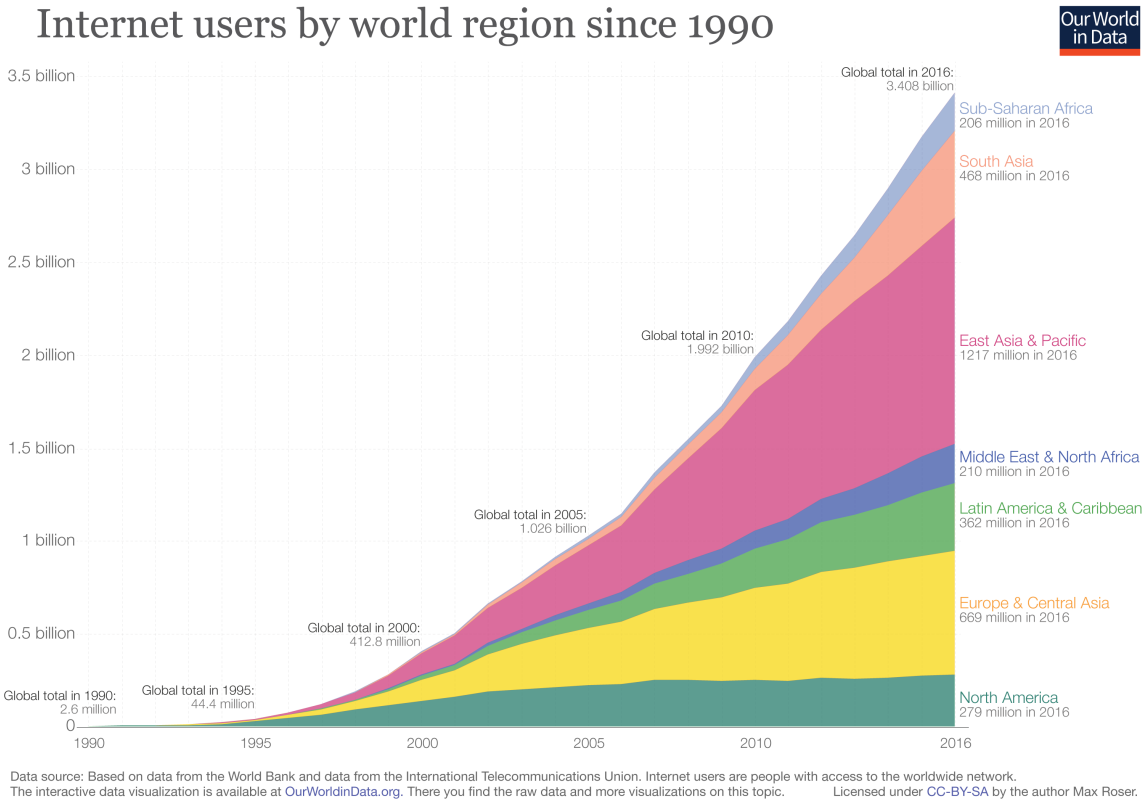


Figure 2: Internet users by world region since 1990. Source: Our World in Data - Based at the University of Oxford[0]

fields such as Green Networking, in which the ability of tools to predict RES is preponderant and very helpful in energy management. For instance, the availability of these tools can allow implementing strategies in the allocating resource on demand (RoD) for power saving in Green Networking context, or in general in the Smart Cities, thinking about to the building's energy consumption and the need of energy management.

The prediction of energy generation is a conceptually easy task, while tools able to do that are hard to implement for computational and modelling reasons. The increment of computational hardware capacity allowed to innovate the state-of-the-art of predicting algorithms, since they require a high computational cost. A tool for predicting energy generation can be expressed in different forms: mathematical, sta-

tistical, or numerical. While the first two forms are the usual one (mathematical or statistical model, such as ARIMA or Linear Regression), the last one exploits the information contained in data self and the Neural Networks are the best candidate in this respect.

The objective of the thesis is to introduce Neural Networks as a tool for predicting renewable energy generation, giving a technological boost in the energy field. It mainly focuses on Photovoltaic energy and in predicting hourly values. To accomplish this hard task, several neural networks were implemented and trained through solar irradiance data, with an hourly and half hour time granularity. They were trained using different datasets, so as in different locations, focusing the analysis in Turin and Buffalo to investigate the network's ability based on different weather locations. Finally, the thesis is structured as following: Chapter 1 introduces the used databases for achieving the objective, the data processing and the comparison with two different databases so as different location, considering Turin and Buffalo; Chapter 2 includes the theoretical part about Neural Networks, their usage in prediction and the different considered techniques of training; Chapter 3 and 4 are the core of the thesis, in which results, performances, and analysis are discussed and justified. At the end, conclusions about the work and possible applications are discussed.

CHAPTER 1

Databases and Data Processing

In this chapter will be explained which data were used for predicting global solar irradiance, from where they were downloaded and which variables they exposed. In addition, there will be discussed the pre-processing on the dataset and the created variables to get better predictions.
There were used mainly two databases: PVGIS (Photovoltaic Geographical Information System) and NSRDB (National Solar Radiation Database).

1.1 Useful definitions

In the following sections some technical terms will be used and below their definition is reported:

- **Solar Irradiance**. It is the solar power falling into a surface per unit area and unit time. It is expressed in W/m^2;
- **Direct Solar Irradiance**. It is the fraction of the solar radiation that reaches the ground without being attenuated by the atmosphere (W/m^2);
- **Reflected Solar Irradiance**. It is the solar radiation that reaches the ground after being reflected by the atmosphere and is considered to arrive from the whole skydome (W/m^2);

- **Diffuse Solar Irradiance**. It is the reflected radiation from the ground surface or nearby obstacles (W/m^2).

1.2 PVGIS database

PVGIS is an online free solar photovoltaic energy database. It allows to download simple solar irradiance, avoiding to use a PV system for energy production, detaching the work from a particular PV system. Most of the solar irradiances contained in PVGIS database are based on the satellite data, which are used to estimate the solar radiation arriving at the earth surface. This calculation approach is justified by the higher accuracy than the sensors ground measurements [7]. The used methods to calculate the solar radiation from satellite have been described in several papers [2] [3] which show the calculation procedure to obtain the solar radiation in a certain region with a given spatial resolution.
Therefore, several types of data can be selected from the database, giving the chance to choose several parameters:

- the location, by inserting the latitude and longitude;
- the solar radiation database (ERA5, SARAH, COSMO, CMSAF), whose differs in the range of data available and algorithm;
- the time granularity (Daily, Hourly, Monthly);
- the radiation components (Reflected, Diffuse, Direct solar radiation);
- the range of the year.

The global solar irradiance and its components (Direct, Diffuse, Reflected) are plotted in Figure 1.1. It depicts Turin hourly solar irradiance in several weather scenarios. Global solar irradiance is defined as the sum of Direct and Diffuse irradiance. The reflected irradiance is usually insignificant compared to direct and diffuse (see Figure 1.1). At the end, to get the global solar irradiance the equation is the following:

$$Gi = Di + Bi \times \cos z \tag{1.1}$$

where z is the solar zenit angle at the time measurements [8]. In this context, for

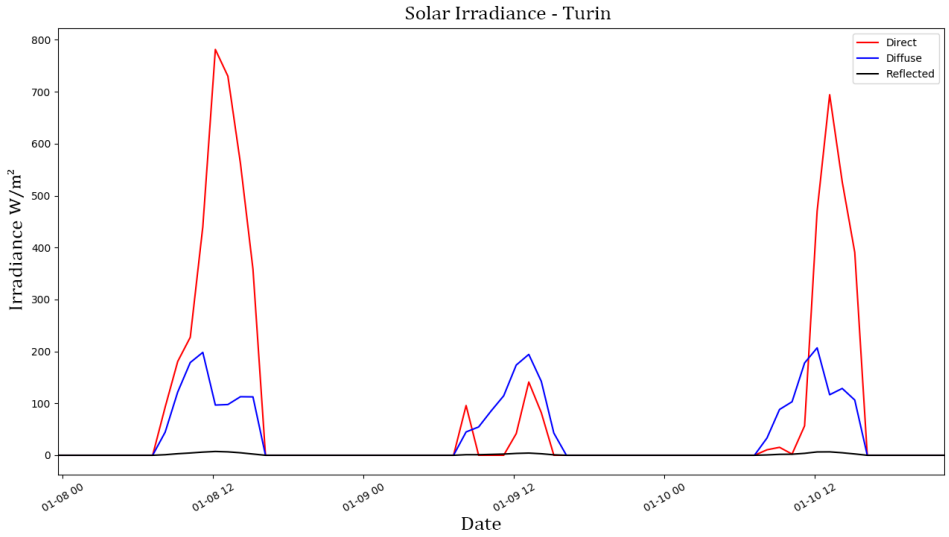


Figure 1.1: Solar Irradiance of three different days in Turin, showing the evolution of the irradiance in several situations (sunny, overcast, or rainy days)

predicting the global solar irradiance all the components were retained, in order to exploit physical correlations between solar variables and by making the prediction detached from any PV system, which should take into account several variables, such as the kind of module and its own *efficiency*.
From the database is possible to obtain PV power production by inserting slope, PV technology, installed peak PV power [kWp], and system loss [%]. However, the power production from the solar radiation can be computed with the following equation:

$$P = Gi \times \sin\theta \times A \times \eta \tag{1.2}$$

where

- Gi is the global solar irradiance;
- θ is the angle of the PV system with respect to the horizontal plane;

- A is the area of the PV system;
- η is the efficiency, that is the portion of energy in the form of sunlight that can be converted via photovoltaics into electricity by the solar cell.

1.3 NSRDB database

The National Renewable Energy Laboratory (NREL) provides solar resource data for the United States through the NSRDB (National Solar Radiation Database) [9], and it is also expanding toward other locations [10]. For the research, the same locations for both databases, PVGIS explained in section 1.5 and the NSRDB, were considered to understand how machine learning techniques behave with respect to two different datasets.
Indeed NSRDB, compared to PVGIS dataset, contains a lot of variables, which are shown in Table 1.1. It has plenty of data since NREL uses PSM (Physical Solar Model) which exploits many algorithms to get much more information. For instance, PSM exploits properties from the satellite retrievals and then uses those properties to calculate surface radiation. Then, to generate cloud properties and precipitation water vapor and other useful features, it uses different algorithms and mathematical model explained in [11]. For these reasons, NSRDB database has more variables and it is richer of information. The utility of this information will be investigated in section 3, where a selection futures is discussed to understand which variable is more effective in predicting global irradiance values.

1.4 Comparision between PVGIS and NSRDB databases

PVGIS and NSRDB databases are obtained using different algorithms for calculating solar radiations. Those algorithms have a significant, so as quantitative, effect on solar irradiance values. Figure 1.2 reports global solar irradiance for both databases,

Name	Unit
Clearsky DHI	W/m^2
Clearsky DNI	W/m^2
Clearsky GHI	W/m^2
Cloud Type	Unitless
Dew Point	Degree C
DHI	W/m^2
DNI	W/m^2
GHI	W/m^2
Fill Flag	Unitless
Snow Depth	Meters
Solar Zenith Angle	Degrees
Temperature	Degree C
Pressure	Millibar
Relative Humidity	Percent
Precipitable Water	Millimeter
Wind Direction	Degrees
Wind Speed	m/s

Table 1.1: All variables of NSRDB

considering as location **Buffalo, NY**.

For comparing them the same range of years 2005-2015 was chosen and for each month the monthly global solar irradiation was calculated. It is evident how the evolution of global irradiance is really different, above all in last months such as November or December of each year. The NSRDB database has also a lower variance (see Table 1.2) respect to the PVGIS and this difference is projected in the prediction phase, in which results are pretty different.

Dataset	CV
PVGIS	1.55
NSRDB	1.49

Table 1.2: CV: Coefficient of variation of both dataset

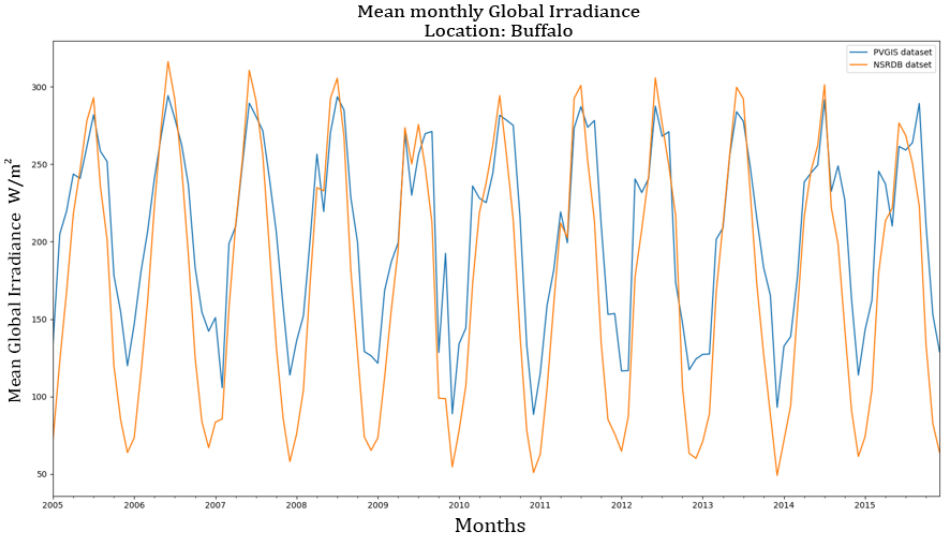


Figure 1.2: Comparing NSRDB and PVGIS dataset

1.5 Data processing

After having analysed which data were available from both databases, data were elaborated and joint in several ways in order to obtain the training and testing set for the Neural Network learning process. The process is explained in sections 1.5.2 and 1.5.3.

1.5.1 Pre-processing

Dataset were normalized in order to reduce the high internal variance, making data more regular. Equation 1.3 was used to normalize data:

$$y = \frac{x}{x_{max}} \tag{1.3}$$

in order to have data between 0 and 1.

1.5.2 PVGIS dataset

The dataset analyzed was the one with a hourly time granularity, as said before. Once the location was chosen, from the PVGIS database two kinds of files were downloaded.
The first one composed of following variables:

- Directed or Beam solar radiation (Bi) (W/m^2);
- Reflected solar radiation (Ri) (W/m^2);
- Diffuse solar radiation (Di) (W/m^2);
- Wind speed at 10 meter (m/s);
- Sun elevation ($deg.$);
- Ambient temperature (°C);

therefore, including all the radiation components. The second dataset is composed of:

- Global solar radiation (Gi) (W/m^2);
- Wind speed at 10 meter (m/s);
- Sun elevation ($deg.$);
- Ambient temperature (°C);

of the same location.
The reason why the Gi was downloaded and added to the first dataset, by considering the same range of years, is due to the high correlation between it and both direct and diffuse variables, as seen in section 1.2. Consequently, both datasets were merged to exploit the correlation between all variables. Figure 1.3 shows the linear correlation between the variables of the dataset, which is consistent with what said about Gi. It is evident how the global solar radiation is highly correlated with the related

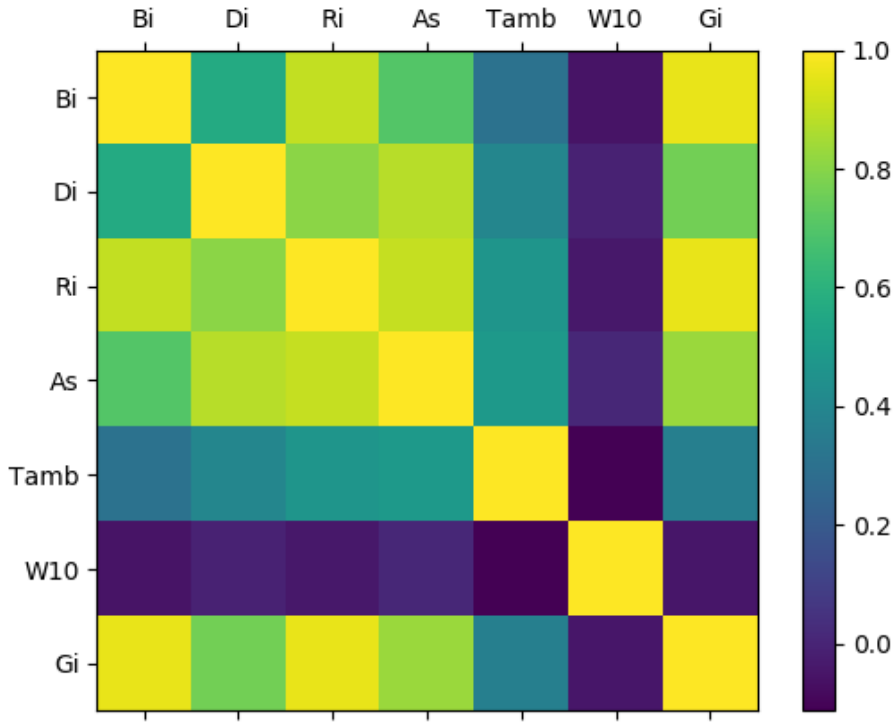


Figure 1.3: Correlation between variables of merged dataset

components Bi, Ri and Di so as the sun elevation As, while the wind speed at 10 meter is not. Moreover, there is a little correlation with the temperature and the solar radiation components.

Since the lack linearity of some variables, all variables were retained to consider for following steps, trying to exploit the capability of a neural network to understand some non-linear dependencies, investigating the possibility to have higher accuracy in predicting. This approach is also analysed in section 3.3.1.

To these variables, the other three were added: Day Of Year[1], Hour, and Month. Table 1.3 shows the final dataset used for training and testing phase, while Table 1.4

[1]It is included in a range between 1 and 365/366

Features	Description
Di_t	Diffuse Irradiance
Bi_t	Direct Irradiance
Ri_t	Reflected Irradiance
$Tamb_t$	Ambient Temperature
$W10_t$	Wind Speed at 10 meter
As_t	Sun Elevation
Gi_t	Global Irradiance
DayOfYear, Hour, Month	Current day, hour and month at time t

Table 1.3: Training set at time t

Feature	Description
Gi_{t+1}	Global Irradiance at time $t+1$

Table 1.4: Value used form the the neural network to learn in predicting

the target values used for calculating the MSE (see equation 2.3 in section 2.1) and backpropagate the error to the neural network during the training phase.

1.5.3 NSRDB dataset

Compared to PVGIS database, NSRDB database allows to obtain two datasets with different time granularity, 30 minutes and hourly. Both types were used and the same steps of PVGIS dataset were performed, explained in section 1.5.2. Thanks to the shorter granularity of NSRDB data was possible to investigate how neural networks with different behave different data granularity. In section 3 will be illustrated this difference between predictions with different time granularity. Obviously, in this case, with a more granular time resolution, it is expected to have better performance in predicting.

In this chapter datasets which will be used for training and testing phase were defined. The correlation between variables pointed out which variables can contribute to predicting global solar irradiance due to their linear correlation. Other variables were added, concerning the day of the year, the hour of the day, and the month. These

variables introduce additional information, allowing improvements in the prediction task. Moreover, the two kinds of databases expose not only a different amount of variables but also different quantitative information, as seen in section 1.4.

CHAPTER 2

Metrics, Neural networks and training approaches

Neural networks are not an easy tool and in this chapter will be explained the procedure to understand which is the most suitable number of hidden layers, and the number of neurons for each layer in this context. Indeed, the design is the most difficult task and a rule of thumb does not exist. Therefore, starting from a general Neural Network overview, the chapter goes in details in explaining how the prediction works, which are the two NNs considered and how they were differently trained. Metrics for evaluating the used training approaches are defined.

2.1 Metrics

Neural networks are usually evaluated through metrics which calculate the error between the predicted and the real value. For the sake of clarity, metrics consider the variables as follows:

- y_i and $\hat{y}_i$ are respectively the real and the predicted global solar irradiance, in which each i-th sample is the value of global solar irradiance at time t;

- $\bar{y}$ is the mean of the real values of the same time period;

- C_{ij} and C_{ii} are respectively the covariance and the variance of real and predicted value.

Below are illustrated metrics used for knowing the goodness of a neural network to predict global solar irradiance values:

- Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{2.1}$$

- Normalized RMSE (NRMSE):

$$NRMSE = \sqrt{\frac{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\bar{y}}} \tag{2.2}$$

- Mean Squared Error (MSE):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.3}$$

- Mean Absolute Error (MAE):

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2.4}$$

- Correlation coefficient (R):

$$R = \frac{C_{ij}}{\sqrt{C_{ii} \times C_{jj}}} \tag{2.5}$$

2.2 Neural Networks

A Neural network is a set of machine learning technique used for several purposes: classification, regression, natural language processing, and prediction. Thanks to its characteristics it is able to catch non-linear correlation between input variables during the training phase, respect to linear techniques. It consists of three parts: neurons, activation functions, and bias. While in the next chapter will be analysed the effective capability of the neural network in predicting future values, mainly global solar irradiance, by using the metrics the mentioned in section 2.1 for evaluating it, in this section will be shown two different architectures of neural network, stressed in several ways. For both techniques the first step was to choose the number of input neurons, the number of hidden layer and the neurons for each of them[1]. This phase took into consideration the PVGIS dataset, considering as location Turin and as range of year from 2005 to 2016. The dataset was split into training set and testing set. The first one included samples from 2005 to 2015, whereas all the samples 2016 as testing set. This approach was called *Yearly training* and will be explained in details in section 2.3.

2.2.1 How the prediction works

The prediction was performed in the following way: a neural network takes a training set composed of hourly features as input (Table 1.3), concerned variables at time t, and it is trained to predict global irradiance at time $t + 1$ (Table 1.4).
Consequently, the network has all useful information about value at time t, so as the same value that it tries to predict but at the hour before. The procedure is iterative. At each iteration, called **epoch**, the network takes a bunch of entries, named **batch** and arbitrary set, and it predicts global irradiance of time $t + 1$ for a given number of epochs. The number of epochs is crucial because a little value does not allow to the network to learn enough, while a higher value from one side could let the

[1]For sake of clarity, some decisions, such as the number of hidden layers or the maximum number of tested neurons were taken small due to computation limit of the Personal Computer.

network learns more about the given training set; on the other side, it could induce *overfitting*[2] problem. At each iteration a loss function is used to calculate the error between the target value (Table 1.4) and the predicted one. Equation 2.3 was the metric used during the training step, therefore adopted as the loss function. The loss is backpropagated so that the network can update its weights and then reducing loss.

In this work Adam Optimizer [4] was exploited because of its simple and computationally efficient characteristics for gradient-based optimization of the stochastic objective function. Indeed, the aim of the Adam Optimizer is to minimize the residual of the loss function

$$\mathcal{L} = MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

After having completed the training phase, the model was obtained, with the right weights and biases. Consequently, it was tested and evaluated with new data, obtaining the prediction. In the following sections will be discussed the approach on designing two kinds of neural networks, a Feed-Forward neural network, and Recurrent neural network. After a first analysis they will be trained, compared and discussed in section 3.

2.2.2 Design methods

The design of a neural network is not so easy. It is straightforward to decide the number of neurons for the output layer since they depend on the outcome that is looking for. By contrast, the number of neurons for the input layer and, above all, about hidden layers is not so easy. On the one side, it is used to take the number of neurons in the input layer as the same number of input variables, even if it is not a rule. On the other side, it is not possible to sum up the design process for the hidden

[2]That is the capability of an algorithm to understand and predict the training data almost perfectly, failing to predict a good value with an input data that the algorithm has never seen before. Whenever it happens, it is used to say that the algorithm is not able to generalize.

layers with a few simple rules of thumb [12].
The used method for choosing the right number of neurons in the hidden layers was the *trial and error* method. Obviously, this method was time-consuming, but it gave a clear idea of how the number of neurons affected the accuracy in both training and testing phase. The equation 2.3 was used for evaluating the error and for choosing the best number of neurons, that is the number of neurons in which the minimum value of MSE is reached for the training and testing set.

2.2.3 Feed-Forward neural network

A feed-forward neural network (FNN) is a simple neural network, which means there are no loops in the network - information is always fed forward, never fed back (Figure 2.1).

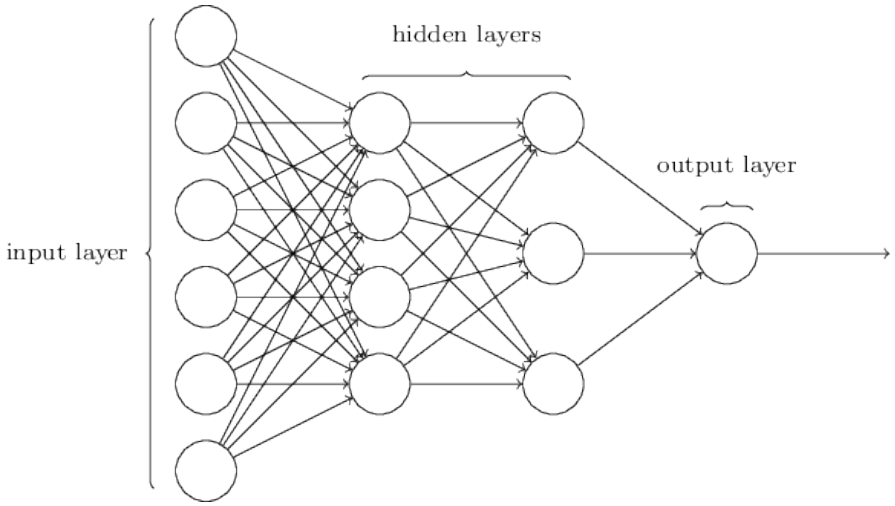


Figure 2.1: Feed-forward neural network

It was designed as follows:

- **Input layers**. The number of neurons was the same as the number of features in the dataset. If the number of features changed in trial and error step, the number of neurons changed accordingly;

- **Output layers**. Since the aim of the thesis is predict the value of produced energy at $t + 1$, knowing the values at time t for each hour, the number of neurons of the output layer is only one;

- **Hidden layers**. For the number of hidden layers, only one layer was retained to minimize the complexity of the network and reducing the training phase, while for choosing the number of neurons for the hidden layer several trials were performed. From figure 2.2 is shown how with a little number of neurons the

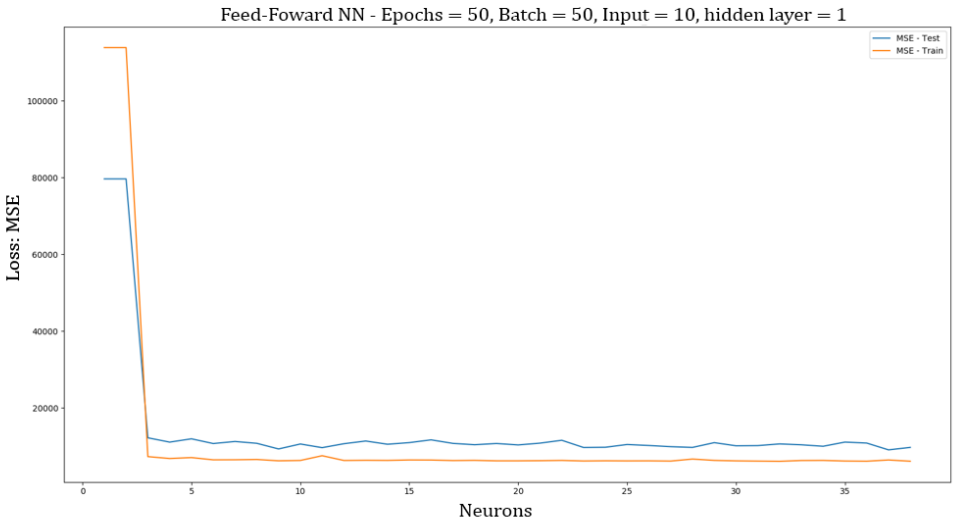


Figure 2.2: Performance of FNN at varying the number of neurons

loss function tends to be constant, meaning that there is not any improvement by adding other neurons. Consequently, in order to minimize the complexity of the network, 15 neurons have been retained.

The employed activation function in predicting the produced energy was the Rectifier Linear Unit or ReLU (Figure 2.3):

$$f(x) = x^+ = \max(0, x) \tag{2.6}$$

It has been proved that the function 2.6 allows to reduce problems in backpropagation steps, such as the possibility to fall in the vanishing gradient issue [12].

Finally, the FNN used in prediction was composed of:

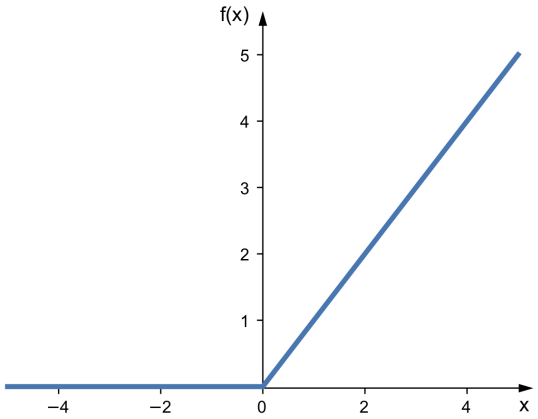


Figure 2.3: Rectifier Linear Unit

- 1 **input layer** with 10 neurons;
- 1 **hidden layer** with 15 neurons;
- 1 **output layer** with 1 neurons;

which was used for the several training approaches explained in section 2.3.

2.2.4 Recurrent neural network

A recurrent neural network (RNN) is a type of neural network used in modeling and prediction of sequential data where the output is dependent on the input. It has been used in applications such as image processing, sentiment analysis, language translation, and speech recognition. RNN has two main differences respect to the FNN:

- the presence of loops in the network;

- an internal memory, which is able to store information about previous calculation.

Indeed, the information keeps changing dynamically an RNN. For example, the behavior of hidden neurons might not just be determined by the activation in previously hidden layers, but also by the activation at earlier times. Therefore, the neuron's activation might be determined in part by its own activation at an earlier time. Thanks to this capability, an RNN is particularly useful in analysing data or in processes that change over time [12], fitting perfectly with the prediction of global solar irradiance.

RNN architecture: LSTMs and GRUs. RNN can be designed in several ways, but the more interesting thing, and also difficult, is the choice of the internal memory architecture. Mainly there exists two architectures, with a **Long short-term memory units** (LSTMs) or with a **Gated recurrent units** (GRUs). LSTM and

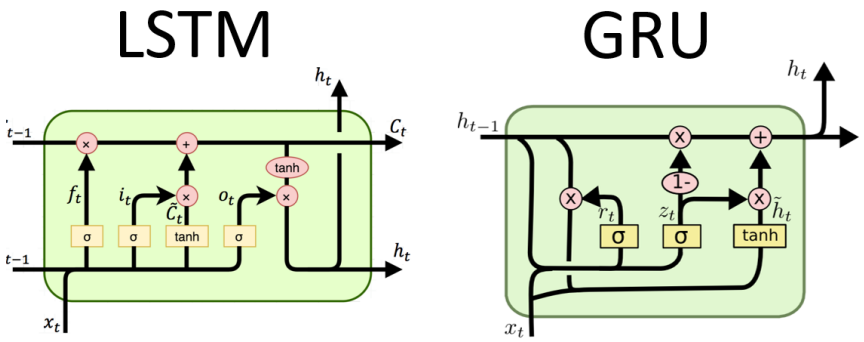


Figure 2.4: LSTMs and GRUs architecture

GRU are shown in Figure 2.4, which is a practical representation of internal architecture of a single neuron inside the RNN. The sequence of activation functions allows to a single neuron (also called *cell*) is able to store earlier information and then taking them into account for the next activations. The LSTM has more gates than GRU, introducing a behavior of *Forget and Output*, in which the cell decides which values maintain and which it sends to the output. This behavior has higher

computational cost, above all thinking about a big number of neurons in a hidden layer. On the other hand, GRU has the characteristic called *Update*, thanks to which decides whether to pass the previous operations or not.
Since both architectures introduce mathematical operations which involve higher computational cost, it was decided to take the simpler one - Gated Recurrent Units - in order to reduce the cost and to speed up the training phase. It has been proved that the performances of architectures mentioned above are pretty the same for several applications [5].

Designing RNN. The design of the RNN followed the procedure explained in section 2.2.2. The number of the input and output neurons remained the same of the FNN, and the number of hidden layers was arbitrary chosen equal to one, because of increasing computational cost at a higher number of the hidden layers. Therefore, with the *trial and error* approach the number of neurons of the hidden layer was chosen, testing the RNN with several numbers of neurons. Figure 2.5 shows the

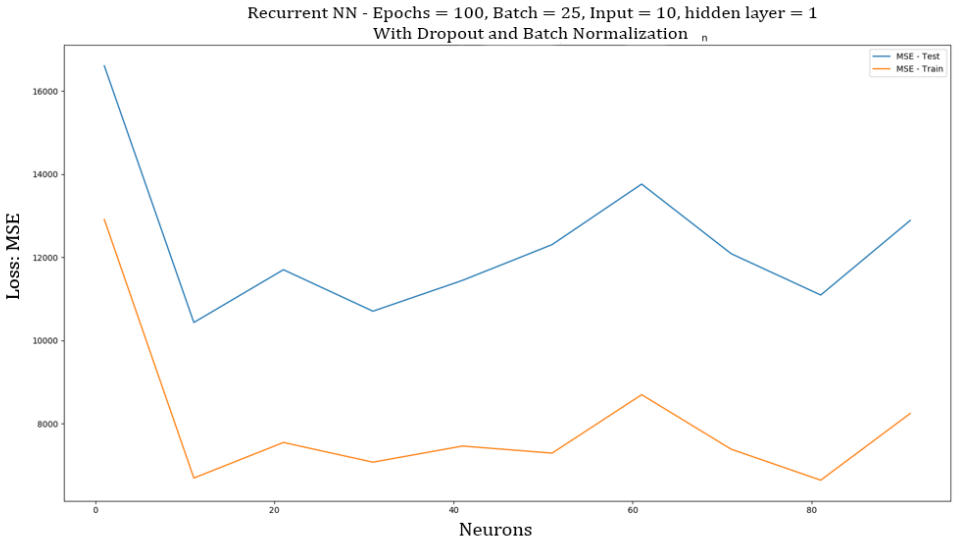


Figure 2.5: Performances of the RNN at varying the number of neurons

performance of the RNN at varying the number of neurons of the hidden layer. In

the x-axis, there is the number of neurons, while in y-axis there is the MSE for each bunch of neurons. Also, two values of MSE are displayed, one for the train and the other for the validation set. Curves are pretty similar and both follow the same evolution. Initially, the error is high for both datasets and then it tends to decrease. At the end, it is possible to notice how the distance between the two curves tends to be larger, with a possible sign of overfitting.
From this figure, the best number of neurons for the hidden layers was set up to 11, in which the MSE is the lowest value for each dataset (training and validation set)..

Dropout, Early stopping, and Batch Normalization. Since an RNN is more prone to the overfitting, several techniques in literature allow to reduce it. In this context the following three techniques were used [12]:

- **Dropout**. It is a technique which radically modifies the network. Essentially, it consists of deleting some neurons over a mini-batch and updating weights and biases. Then, the network is restored and other neurons are randomly selected and deleted. The procedure continues until whether it converges or the number of epochs is met.

- **Early stopping**. It is used to determine the minimum number of epochs for training a neural network. At the end of each epoch, a part of the dataset is used as a validation set with which MSE is computed. Whenever the MSE does not improve anymore over the epoch, the training phase is earlier stopped and so the training phase is terminated.

- **Batch Normalization**. It allows to increase the stability of the neural network and reducing the internal covariate shift, normalizing the output of a previous activation layer. Indeed, it subtracts at each batch its mean and dividing for its standard deviation. The technique considers to perform the normalization for each mini-batch, and backpropagate the gradients through the normalization parameters. In the end, these techniques ensure a faster training step [6], useful for speeding up the training phase of an RNN.

At the end the final configuration of RNN was:

- 1 **input layer** with 10 neurons
- 1 **hidden layer** with 11 neurons
- 1 **output layer** with 1 neurons

2.3 Training approaches

The way in which a neural network is trained can significantly affect in reducing errors because it can interpret information differently on the base of the way it sees data. Consequently, changing the training approach also the amount of error changes during the training phase. In this section, there will be explained several training methods, in order to understand whether a method can perform better than others or which could be the best in some scenarios.
The method used to train data were the following:

- *Yearly training.*
  The yearly training is the simplest method used. It consists of taking a range of years as training set and another range as testing set. For instance, the dataset discussed in section 1.5 included years from 2005 to 2016, and the training set is composed of years from 2005 to 2015, while it is tested with the last year, 2016. Finally, there is a single neural network for predicting hourly values, for all months.

- *Monthly training.*
  Differently, with the previous approach, a monthly training consisted of creating as neural networks as months are. In this case, 12 NNs were trained, one per each month. For example, from a range of the year 2005-2015, all the January are taken as training set; the January of 2016 is used as testing set. In this way, the neural network specializes in predicting values in a given month. It will be shown that for some months this approach performs better than other methods.

- *Seasonal training.*

  The last approach follows the intuition of the monthly training, but instead of considering a single month the dataset was split in 4 other datasets, one per each season. Instead of the astronomical seasons, which are the usual ones, in this context, the meteorological seasons were considered, since they are reckoned by temperature and fits better with the aim of the thesis. Meteorological seasons are different for the northern and southern hemisphere. For the northern hemisphere, spring begins on 1 March, summer on 1 June, autumn on 1 September, and winter on 1 December. For the southern hemisphere, spring begins on 1 September, summer on 1 December, autumn on 1 March, and winter on 1 June. In the end, 4 datasets were obtained and then 4 neural networks were trained, one per each season. The training set included all the months which belong to a season, considering the range of year from 2005 to 2015, Then, a season of the last year (i.e. winter 2016) was considered for testing the relative seasonal neural network.

The theory beyond the neural network is huge and here only a spot was reported. The difficulties in training and then obtaining good results bring to find new kinds of approaches. In this chapter, three new types of approaches were proposed, exploiting the data characteristics such as seasonality, and similarities between months. Also, FNN and RNN are two different types of neural network in several aspects such as in training and in the designing process. Indeed, RNN is more complex than FNN, becoming time-consuming without the right hardware which can make the difference. Moreover, FNN and RNN can perform very differently if they are fed with a different type of structured dataset, as discussed in the next chapter.

CHAPTER 3

Discussion of results

Once neural networks were designed (2.2.2), several approaches were tested. In the following sections will be shown the goodness of RNN and FNN for predicting global solar irradiance after being trained, using metrics defined in section 2.1 and comparing errors, shapes. Initially, Turin will be considered; then, predictions will be compared with another city with a similar latitude but different longitude, Buffalo.

3.1 FNN vs RNN

As explained in section 2.3, several training approaches were adopted to investigate the capability of a neural network to learn data, information, and non-linear correlation, feeding it in different ways. However, all methods had in common:

- the batch size, set at 50;
- the learning rate $\eta = 0.001$.

3.1.1 Yearly training

Starting with the simplest training methods, both FNN and RNN networks were trained with Turin data, from 2005 to 2015 and tested with the last year, 2016.
Figure 3.1 and 3.2 show the MSE for training and validation set, regarding FNN and

RNN. They show how the network learns after a few epochs. Indeed, for the Feed-Forward the train MSE tends to be constant after 50 epochs, so as the validation MSE, which is pretty near to the train one. Also, with few epochs, any overfitting scenario is shown. This is true for both FNN and RNN. Moreover, the order of error is the same for the two techniques, pointing out that the different architecture does not affect the committed error when the networks are trained with the same sequence of values. However, RNN is able to learn in fewer epochs (figure 3.2) than FNN. While MSE's FNN shows two down picks before seeing a constant value, RNN goes immediately down and then few improvements are gained. Moreover, the validation error is always lower than the training due to the *Dropout* (2.2.2) technique, since the network continuously changes its structure during the training phase. Looking

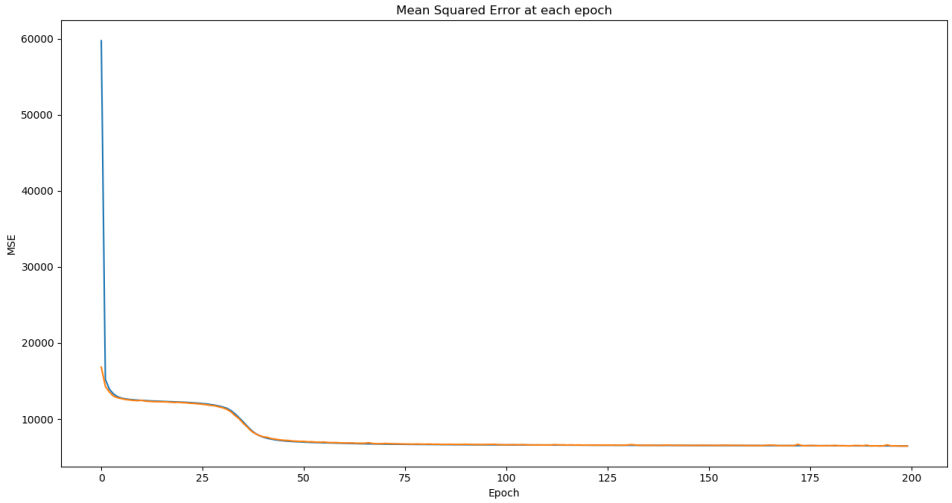


Figure 3.1: MSE yearly training for FNN

at figures 3.3 and 3.4, the prediction of January 2016 is shown and other differences between the two techniques can be found. For instance, the succession of days and night is better followed by the RNN than FNN. This is undoubtedly due to the memory in the RNN (2.2.2), which is able to catch this kind of behaviour from data. On the contrary, shapes are pretty similar between them, the pick is nearly always

underestimated in both methods, except for very bad days. For these days, both networks overestimate global solar irradiance and therefore PV energy production.

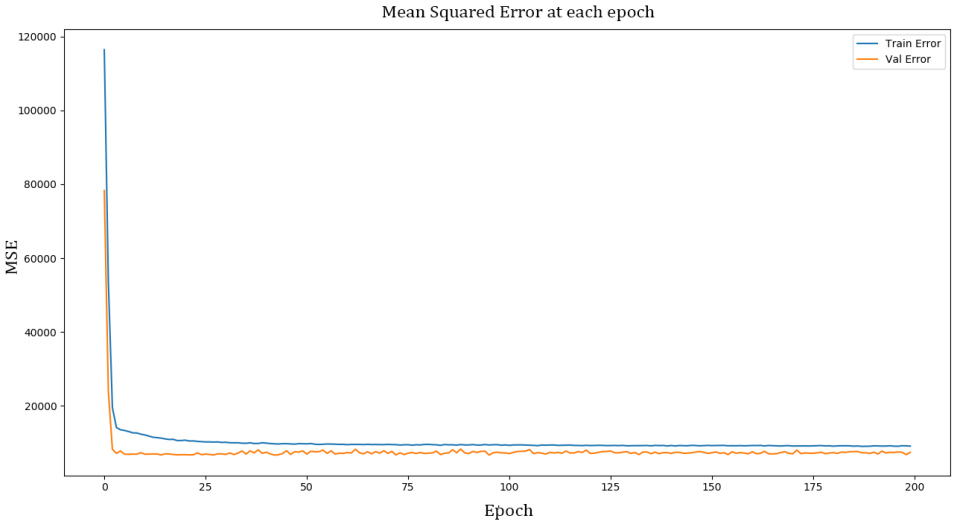


Figure 3.2: MSE yearly training for RNN

	FNN		RNN	
Month	MAE (W/m^2)	MSE (W/m^2)	MAE(W/m^2)	MSE(W/m^2)
January	59.447	18911.647	51.693	16544.160
February	64.919	20646.501	59.725	18794.057
March	44.746	7274.466	37.509	6256.451
April	61.023	14041.393	56.892	12031.990
May	52.878	9137.077	50.169	8088.034
June	53.789	10290.037	51.461	9653.121
July	42.036	7137.668	44.772	7046.896
August	31.588	5278.970	37.480	5592.203
September	37.655	6041.242	38.153	5960.992
October	32.504	5186.052	32.958	5684.040
November	25.501	2870.602	24.862	3040.531
December	63.759	17394.810	56.867	15604.584

Table 3.1: Monthly errors with yearly training, for both RNN and FNN

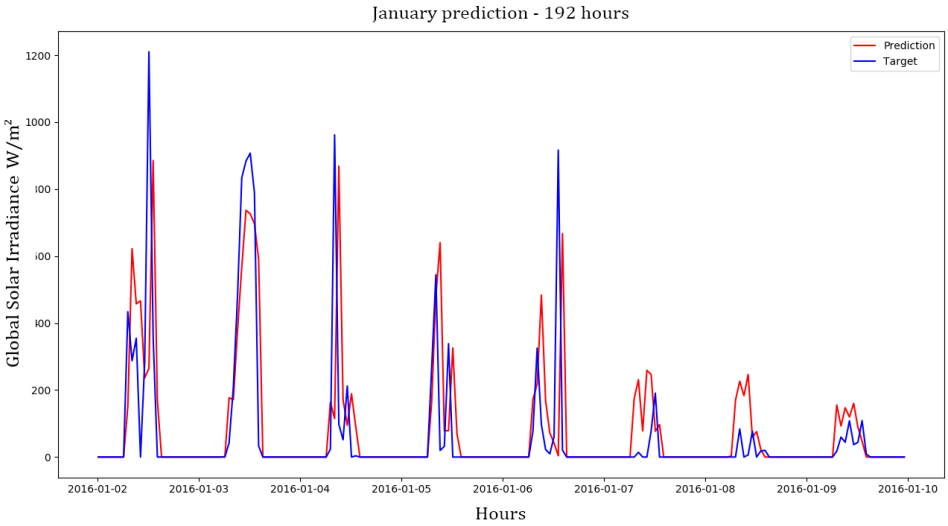


Figure 3.3: FNN prediction - January

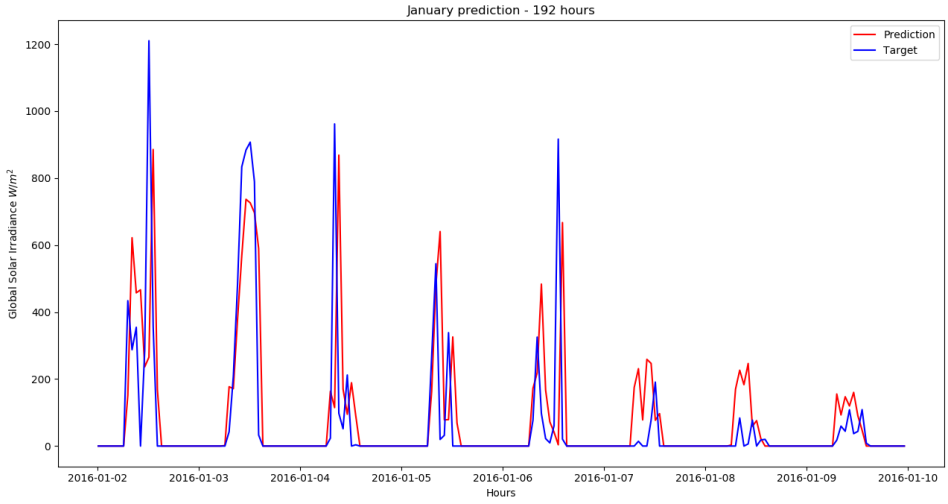


Figure 3.4: RNN prediction - January

Table 3.1 shows values of the metrics explained in section 2.1, in which both networks were trained with the yearly approach and tested in predicting different months. As

expected, RNN performs better than FNN, although not so much. Indeed, looking at these errors it would be expected even better performance from RNN, making it more suitable. The differences between the two networks are not so evident, but as shown in Figure above (3.3 and 3.4) only the shape of predictions has little changes.

3.1.2 Monthly training

The second approach was the monthly training. As explained in section 2.3, the network was trained with all the same months in the training set and tested with the same month of the testing set. January was considered, in order to compare it with the yearly training.
First of all, the training phase was faster than the previous training, since the amount of data were fewer. Figures 3.5 and 3.6 show the MSE for both FNN and RNN. *Overfitting* can be noticed during FNN training phase since the validation MSE rise up after few epochs - around 100 - and continuously increases, even if slowly. This behaviour is not so evident in RNN, but at the last epochs the validation MSE rise reaching the training error. For these reasons the training phase was stopped around 200 epochs - early stopping 2.2.2 - since there were no improvements and the capability of the network to predict got worse.

After networks were trained the prediction was made and the result is shown in figures 3.7 and 3.8, which are pretty different. The FNN performed worse with some non-zero values during the night, even if it followed better the pick in good days, at least where the variability of the day was lower and the energy production would be higher. This behaviour was not seen in RNN, figure 3.8, whose prediction was the same as the previous training approach, and therefore better than FNN with monthly training. Table 3.2 compares errors between FNN and RNN. Looking at the table would be straightforward say that FNN performs better, except for some cases such as the error in predicting November, in which the error is smaller than 10 W/m^2, by the way, the presence of non-zero values so as the shape of the prediction were bad as January. In the end, it can be closed that the monthly training does not

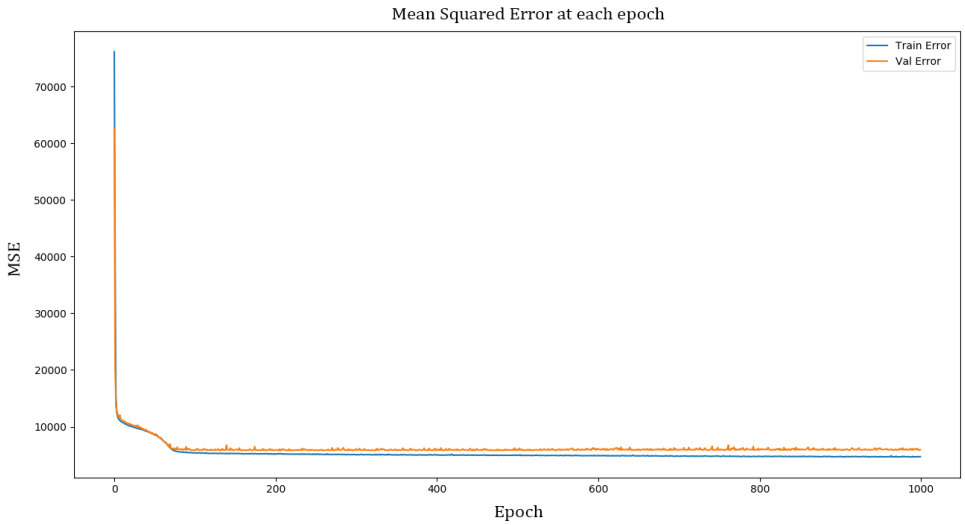


Figure 3.5: MSE monthly training for FNN

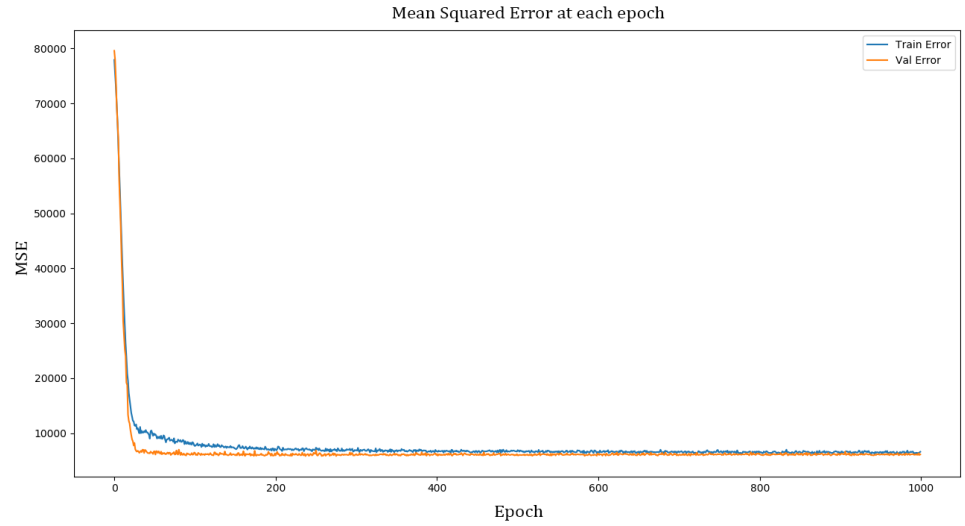


Figure 3.6: MSE monthly training for RNN

improve the prediction significantly, while the FNN performs even worse by looking at the shape during the night.

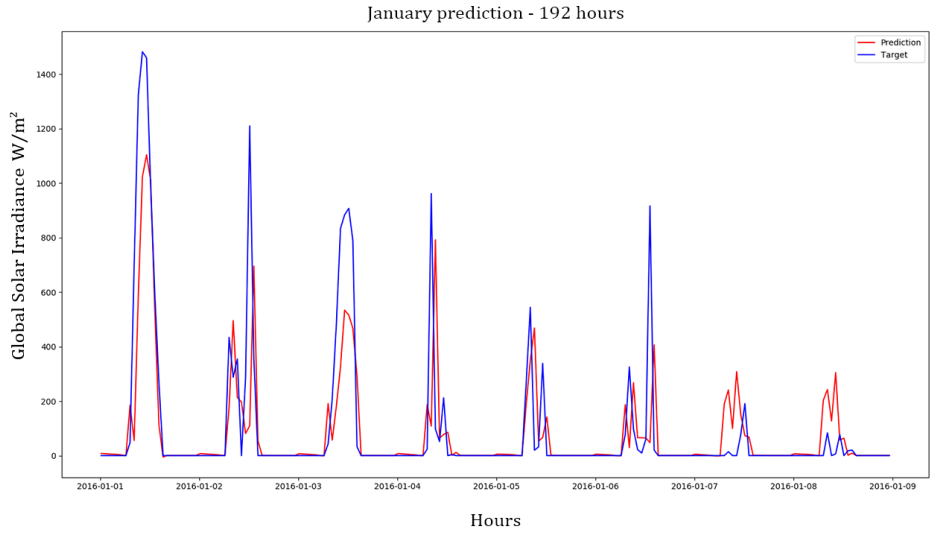


Figure 3.7: FNN prediction - January

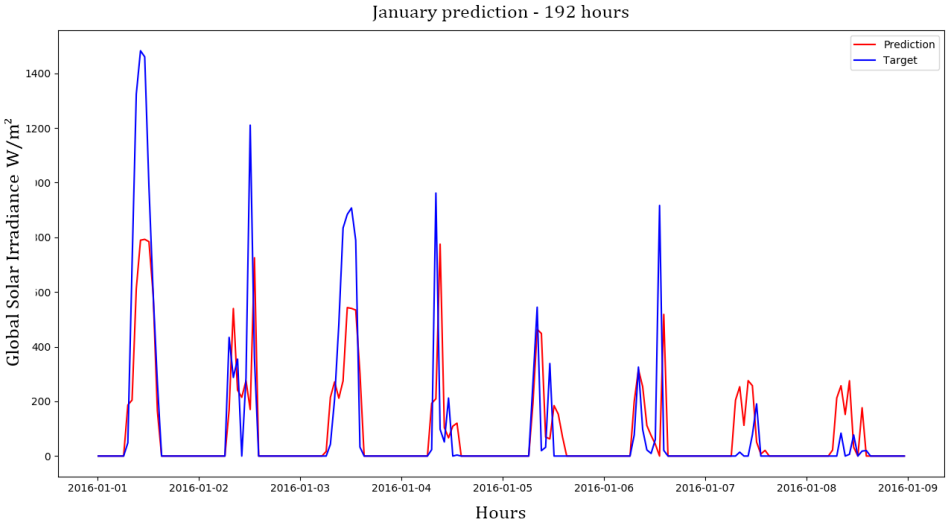


Figure 3.8: RNN prediction - January

3.1.3 Seasonal training

The third approach is the seasonal training, which would have exploited the seasonality of some months to reduce the variance and improve the prediction in months

	FNN		RNN	
Month	**MAE**	**MSE**	**MAE**	**MSE**
January	55.507	19130.39	55.751	18408.4
February	64.638	20863.04	64.597	22460.75
March	34.988	5662.456	37.755	6758.585
April	56.695	12081.59	56.311	12130.96
May	47.475	7684.21	49.039	8017.887
June	46.818	8135.207	47.727	8499.504
July	36.815	6454.579	49.082	7534.211
August	33.193	5574.556	35.26	5942.328
September	34.735	5087.49	40.045	5839.31
October	29.945	4521.132	30.453	5015.937
November	31.472	3870.092	20.408	2349.664
December	62.467	17740.92	60.831	16728.53

Table 3.2: Monthly errors with monthly training, for both RNN and FNN

with similar weather. Consequently, 4 neural networks were gained, one per each season. Figures 3.11 and 3.12 shows the MSE during the learning phase for FNN and RNN accordingly. For both FNN and RNN, the evolution is practically the same as the previous approaches, even though there are three main differences. First of all, looking at the first epochs, the MSE does not go up and down during the first epochs of the learning phase, as seen in the other training approaches. Secondly, the overfitting is not visible, even if around 1000 epochs the validation error increases but does not overcome the training error. Lastly, there is a different shape of the predicted values. In fact, FNN performed very well and it can be seen in figure 3.11, while in figure 3.12 is shown the RNN's predictions. While the RNN did not show any variation in predicting values with this approach, the FNN did. Looking at the shape of the prediction, it seems that was able to follow better the evolution of global solar irradiance. Finally, the table 3.3 shows errors for both RNN and FNN. In some scenarios, such as October and November, the RNN is prone to perform remarkably better than FNN, almost halving the error. Moreover, the MSE results even smaller, a sign that the RNN learnt better.

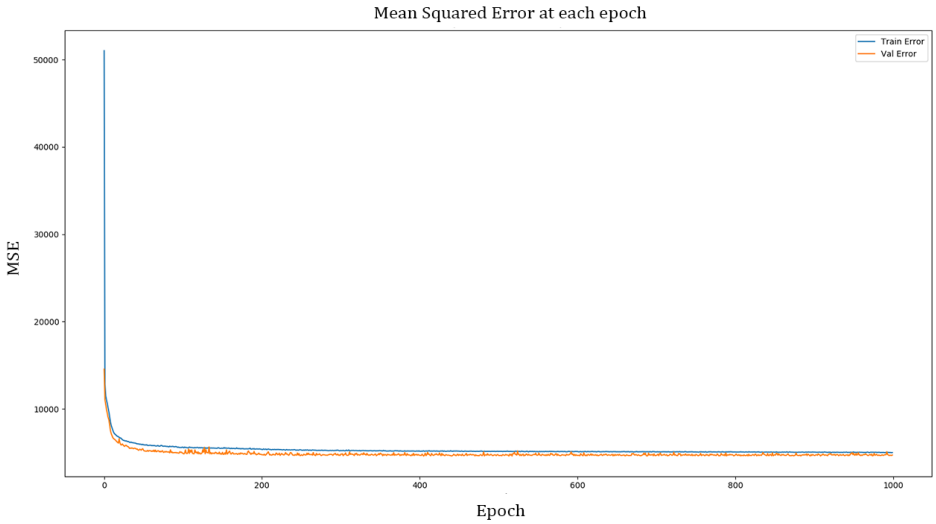


Figure 3.9: MSE in seasonal training for FNN

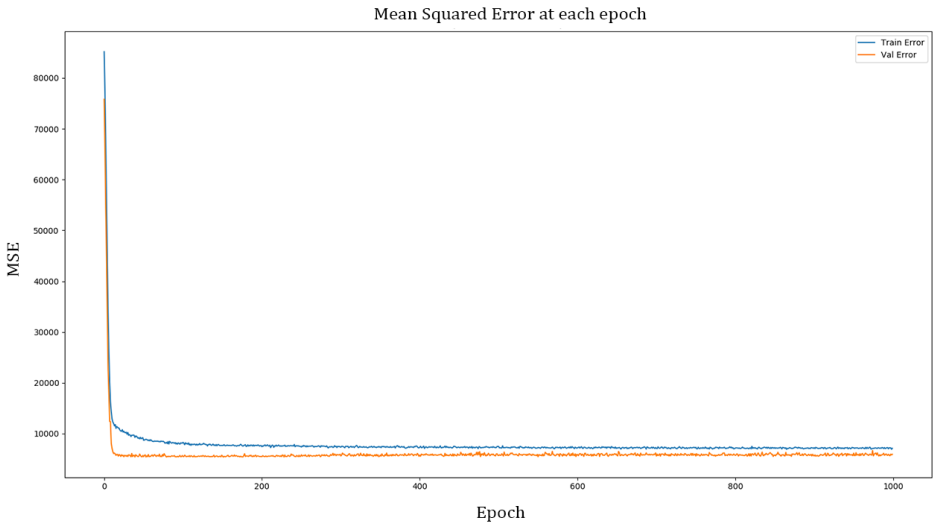


Figure 3.10: MSE in seasonal training for RNN

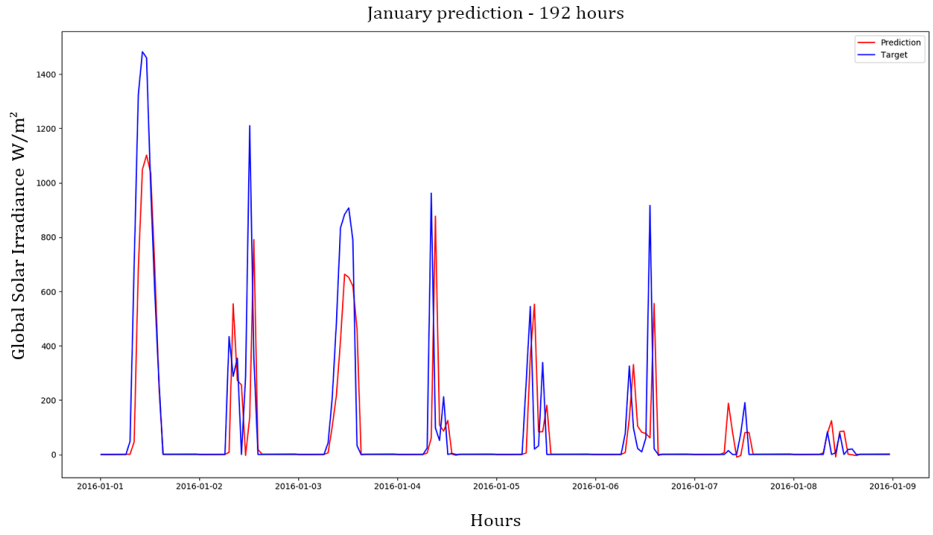


Figure 3.11: FNN prediction - January

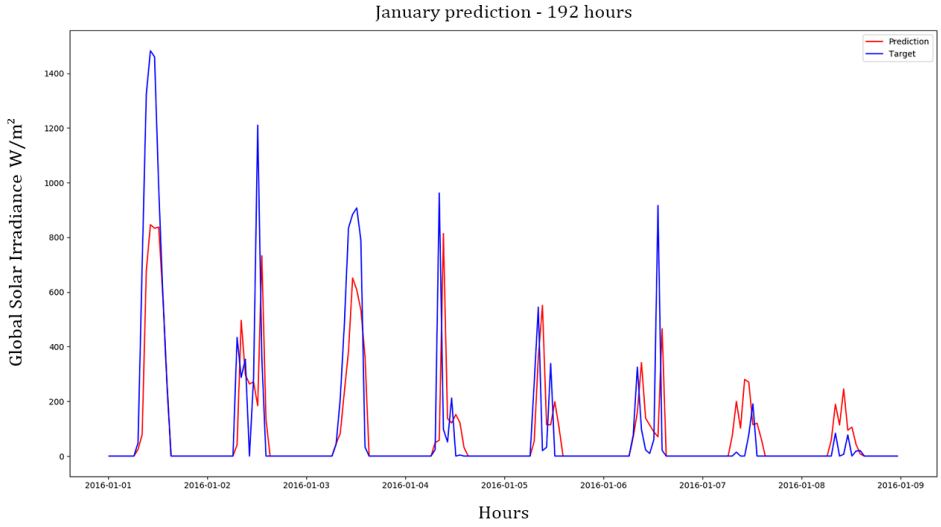


Figure 3.12: RNN prediction - January

3.1.4 The best choice

The analysis above brought significant results whenever a NN has to be chosen for decision making, i.e in management systems. Tables 3.4 and 3.5 depicts errors

	FNN		RNN	
Month	**MAE**	**MSE**	**MAE**	**MSE**
January	55.823	17642.25	51.13	16840.33
February	72.745	24589.76	65.023	21296.29
March	45.652	7309.608	45.564	8241.674
April	59.704	12248.54	54.963	11618.89
May	49.817	8129.039	49.499	8246.942
June	61.089	11970.93	53.599	9672.017
July	46.951	7870.746	48.447	7437.507
August	41.083	7146.45	37.78	6429.089
September	46.215	7838.521	46.731	7880.876
October	52.335	10182.18	34.422	6338.108
November	53.582	10622.9	29.603	3933.827
December	82.653	28609.83	77.75	27471.71

Table 3.3: Monthly errors with seasonal training, for both RNN and FNN

differently, so that can be analyzed the behaviour of a given neural network with the training approach. FNN seems performs well in some cases and the mean MAE is pretty the same as the RNN. This result was not expected since the RNN was the favorite because of good performance with sequential data. On the other side, networks performed differently in predicting, with some errors during the night in catching the pick of some days. Therefore, by looking at the error, no choice could be taken. However, the RNN performed very well in some cases and the mean MAE, in almost all training approaches, is lower than FNN. In some cases, the reduction of committed error with RNN is remarkably smaller than FNN. For instance, by taking into account Autumn and winter months, in which the energy production is not so certain and good, the RNN gained better results. In the end, the best training approach resulted, in mean, to be the yearly training combined with RNN, which gave best results in the worse case of production.

	FNN		
Month	**Yearly training MAE**	**Monthly training MAE**	**Seasonal training MAE**
January	59.447	55.507	57.886
February	64.919	64.638	71.937
March	44.746	34.988	48.249
April	61.023	56.695	56.885
May	52.878	47.475	49.477
June	53.789	46.818	59.854
July	42.036	36.815	49.077
August	31.588	33.193	39.388
September	37.655	34.735	48.709
October	32.504	29.945	51.518
November	25.501	31.472	50.821
December	63.759	62.467	79.537
Mean MAE	**47.487**	**44.562**	**55.278**

Table 3.4: Comparison of monthly errors of FNN for each training approach

3.2 Comparing Turin vs Buffalo

Once both the best network and training approach was chosen, another city was considered to compare the performance based on different coordinates. The city is in the same latitude but at different longitude: Buffalo, NY. Consequently, by using PVGIS tool and the same procedure explained in section 1.5, the new dataset was created. The only difference was due to the range of years available, from 2005 to 2015. To be comparable the prediction between the Turin and Buffalo, the same range of data was considered. Therefore, for both cities:

- the training set included data from 2005 to 2014;
- the testing set included data of 2015;
- the same name of epochs (200) and the same batch (50) were used, so as the learning rate $\eta = 0.001$.

	RNN		
Month	**Yearly training MAE**	**Monthly training MAE**	**Seasonal training MAE**
January	51.693	55.751	51.13
February	59.725	64.597	65.023
March	37.509	37.755	45.564
April	56.892	56.311	54.963
May	50.169	49.039	49.499
June	51.461	47.727	53.599
July	44.772	49.082	48.447
August	37.480	35.26	37.78
September	38.153	40.045	46.731
October	32.958	30.453	34.422
November	24.862	20.408	29.603
December	56.867	60.831	77.75
Mean MAE	**45.212**	**45.605**	**49.543**

Table 3.5: Comparison of monthly errors of RNN for each training approach

Figure 3.13 shows the prediction of first 192 hours of January in Buffalo, while Figure 3.14 shows the prediction in Turin. Figures shows several type of days, so

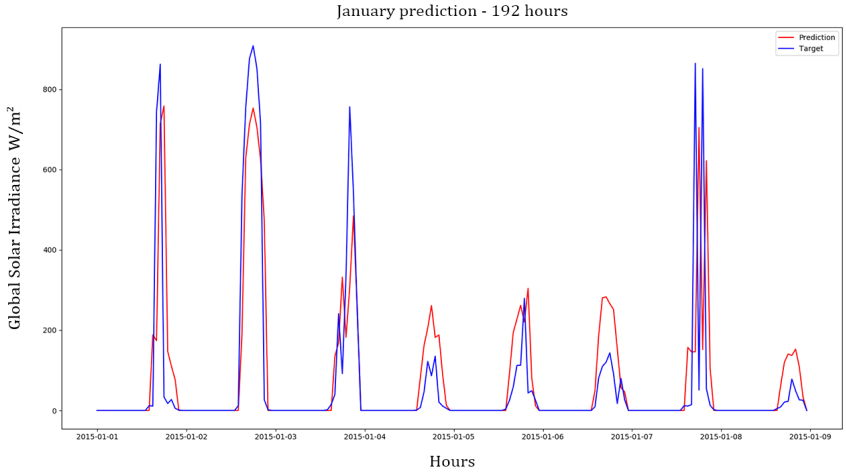


Figure 3.13: RNN prediction with yearly training - January, Buffalo

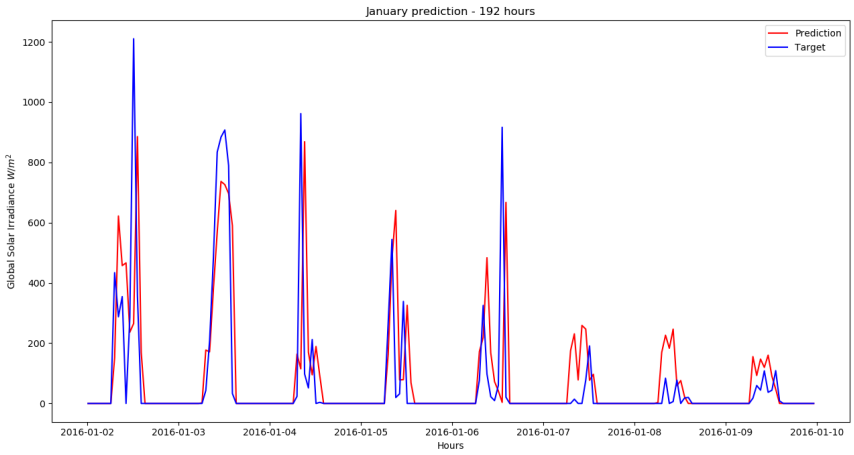


Figure 3.14: RNN prediction with yearly training - January, Turin

that is possible to see the behaviour of the network in different scenario, whenever the irradiance is high, low, constant or highly variable. Substantially, the network behaves in the same way, during the bad days the global solar irradiance is usually overestimated, while during good days, in which the shape of the irradiance is the usual one, the network underestimates it. Consequently, by changing dataset and location the behaviour of the network did not change, demonstrating no relation between coordinates and amount of error. Maybe the chosen locations have the same weather, therefore no difference in training and testing the network was caught.

3.3 NSRDB dataset and selection features

The NSRDB dataset illustrated in section 1.3 contained more features than the PVGIS and in this section will be investigated if it could be a good point or flaws. The availability of more features can bring noise in the training set, causing a raising of error in predicting (this is also true for other scenarios in the machine learning, such as classification, regression, NLP). Below, the approach to a selection the right type of features in NSRDB for the predictions will be shown, looking into the performance

of the RNN at various number and typologies of features. In fact, the interest was in understanding which features could have contributed to reducing the MAE and MSE, considering features shown in table 1.1, section 1.3.

3.3.1 Selection features

The selection feature process involved the creation of many datasets with different characteristics (i.e. mean and variance). In order to be able to compare those datasets, other metrics were used, defined in section 2.1, such as:

- Normalized *RMSE* (equation 2.2), for comparing dataset;
- Correlation coefficient R (equation 2.5), to understand which variable is more effective in predicting global irradiance values.

Therefore, these metrics were added to the usual one so that a better analysis could be done. Table 3.6 reports all the several tested combinations of variables with the NSRDB dataset. This table has to be considered for analyzing the table 3.7, which shows errors associated with each dataset. In this last table is highlighted the best result obtained, which is the 6-th dataset. The main difference between the other dataset is the presence of the *Hour*, indicating the high effect that this variable affects in predicting, as expected from an RNN and from the kind of sequential data. With the 6-th dataset, even the coefficient correlation is higher than others, except for the 9-th which has a very little improvement. These datasets differ from the *Precipitation Water* feature, which could be indicative in a better shape of the prediction. The NRMSE is even smaller in the 9-th than 6-th dataset. It is curious how the Clearsky condition does not affect so much in predicting the global solar irradiance, that is the value of DHI, GHI, DNI in clear sky condition, maybe because they are not so informative and they are redundant combined with the normal condition DHI, GHI, and DNI.
Others effective results were obtained with the 10-th and 11-th dataset. Two different combinations of variables allow to obtain little errors as much as the 9-th.

NRMSE'11-th is smaller than the 9-th, indicating a little improvement as a combination of variables. Finally, new features did not allow to get better results as could be expected. The dataset self improved the capability of the network to predict the best result. This is valuable information since highlight how much the process by which data are collected and processed affects in the predictions. This is better analyzed in the following section.

Dataset	Hour	Clearsky DHI	Clearsky GHI	Clearsky DNI	Cloud Type	Dew Point	DHI	GHI	DNI	Solar Zenit Angle	Temp.	Press.	Humitiy	Prec. Water	W. Dir.	W. Speed
1							x				x	x	x			
2							x	x			x	x	x			
3					x		x	x		x	x	x	x			
4							x	x		x	x	x	x			
5					x	x	x	x		x	x	x	x			
6	x				x		x	x		x	x	x	x			
7					x		x	x		x	x	x	x	x		
8		x	x		x		x	x		x	x	x	x			
9	x				x		x	x		x	x	x	x	x		
10	x				x	x	x	x		x	x	x	x	x		x
11	x						x	x	x	x	x	x	x	x	x	x

Table 3.6: Combination of variables for composing several dataset

Dataset	MAE	MSE	NRMSE	R
1	83.896	22081.759	0.876	0.822
2	61.518	11860.277	0.642	0.908
3	58.003	10809.780	0.613	0.918
4	60.275	11163.412	0.623	0.913
5	59.903	10921.337	0.616	0.918
6	35.356	6332.660	0.469	0.952
7	58.001	10552.458	0.6055	0.919
8	58.081	10833.238	0.6135	0.917
9	35.732	6308.307	0.468	0.953
10	39.720	6804.874	0.487	0.951
11	38.761	6705.77	0.483	0.951

Table 3.7: Errors prediction for each dataset

3.3.2 NRSDB results vs PG results in predicting

To better understand if the new kind of dataset allows obtaining a better result, it was compared with the previous one, PVGIS. Since NSRDB had a wider range of years available and also a bigger number of features. First of all, the same features for both dataset were selected. Then, several cases were tested and Table 3.8 illustrates results in predicting global solar irradiance by using different training sets. The testing set was the next year available for each dataset. Looking at first two rows, differences between PVGIS and NSRDB can be seen. Indeed, NSRDB dataset allows to the network learns better. NSRDB'MSE had even the half of the PVGIS error, considering the same range of year and same type of features. This result can be

Training Set	MAE	MSE	NRMSE	R
PVGIS (2005-2015)	53.953	14231.946	0.558	0.936
NSRDB (2005-2015)	41.006	7045.559	0.481	0.948
NSRDB (1998-2016)	39.037	7119.630	0.497	0.946

Table 3.8: Prediction errors - NSRDB vs PVGIS dataset

explained by the fact that the two datasets are created using different algorithms, and

this result is relevant in perspective of future applications and in predicting values in real use cases. Finally, the last row shows smaller errors values than others. However, the correlation coefficient is not so high than the training set which considers fewer years, so as NRMSE is higher in NSRDB (1998-2016). This can be caused by the higher variance introduced with more years available in the training set.

3.3.3 NSRDB with finer time granularity

Since NSRDB tool allowed to obtain a finer time granularity dataset, the prediction with this kind of dataset was investigated, so that was possible to understand if more information were helpful in predicting irradiance values.
The same range of year of PVGIS (2005-2015) was chosen, and Buffalo as location. In this way the prediction was comparable with other time granularity, by using the yearly training approach. From figure 3.15 is possible to see how the network

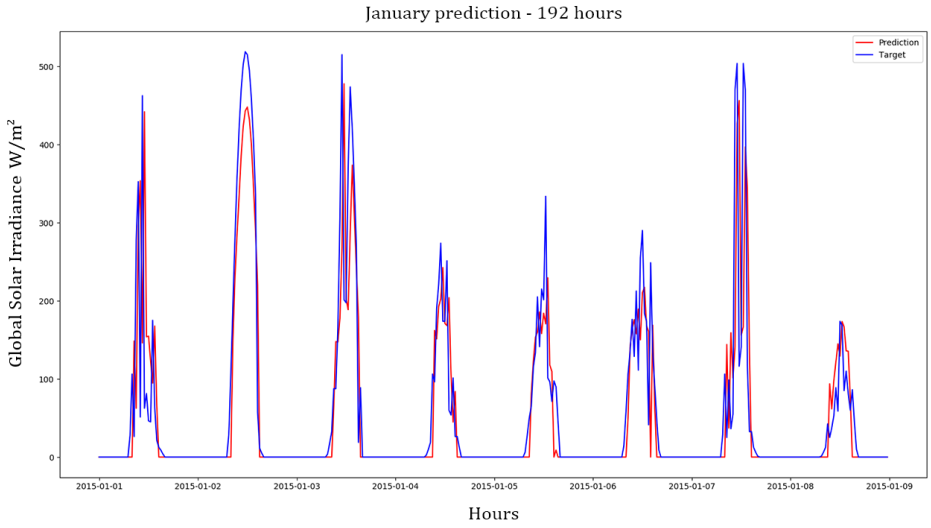


Figure 3.15: January prediction with 30 minute time granularity

performs very good respect to the others approaches, due to the bigger number of features per hours. The network predicted the value of global solar irradiance of the

next hour with values of the previous half hours and this is way it performed even better, following very well the global solar irradiance'shape so as the several picks.

Here several results were discussed, starting from the comparison between the two techniques, RNN and FNN, based on the training approach. This step defined RNN with yearly training as the best technique. This result was obtained because both monthly and seasonal training contained less amount of data so that the neural network was not able to generalize the task in which it was specialized. For instance, whenever a month has high variability during the range of the considered years for training, the monthly training can reduce its accuracy since it has been trained with months with very different values, with consequences in lack of the desired specialization. The same can be said about the seasonal training. Moreover, the location does not affect in training phase, in which the neural network performances did not change. Finally, the NSRDB dataset can allow to boost the performance of neural networks thanks to its available features. The selection features section shown how some variables can improve performance more than others, making the difference in prediction.

CHAPTER 4

Post-processing and cluster analysis

After having obtained a tool with good predictions performance, the thesis moved on matching data analysis with the prediction task to catch other useful information. For instance, it was useful to understand whether the prediction performs well during the first hours of the day, sunrise, or in the last hours, sunset. Moreover, some days seem to be similar to the amount of global solar irradiance and about the sequence of bad days and good days.
In order to extrapolate this information a clustering algorithm was used, in particular K-means, so that groups of days were created and analysed on the base of weather similarities and daily pattern similarities.

4.1 K-means

K-means belongs to the category of *unsupervised learning*, approach in which an algorithm is able to learn or group data on the basis of common characteristics. K-means is a partitional clustering method which aims to divide data space into k clusters. From a set of N observations $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k, ..., \mathbf{x}_N)$ in which $\mathbf{x}_k \in \mathbb{R}^d$, K-means constructs $K \leq N$ disjoint subsets minimizing the within-cluster sum of square, defined as:

$$WSS = \sum_{k=1}^{N} (\mathbf{x}_k - \bar{\mathbf{x}}_C)^2 \tag{4.1}$$

where $\bar{\mathbf{x}}_C$ is the centroid of the cluster C $\in K$, that is the mean value of samples in the C-th cluster, while $\mathbf{x}_k$ is k-th observation of the C-th cluster.
The algorithm works on the base of the following steps:

1) from a set of observation, it picks k random data points and defines them as centroids, which means that the centroid $\bar{\mathbf{x}_C}$ is initially randomly selected;

2) it calculates the distance between data points and every centroid with the equation 4.1. Then, k clusters are defined by considering all data which are closest to a centroid;

3) grouping data in clusters leads to the definition of new centroids. This time the centroid is computed calculating the mean of all samples in the cluster.

Step 2 and 3 are continuously repeated until the centroids stop moving, which means the K-means algorithm is converged.
In the following sections will be analysed several approaches for clustering PVGIS dataset, to investigate whether clusters with the same type of day were possible to obtain, such as sunny, cloudy, or rainy day.
This could be useful for understanding if a day is classified as high or low production so as high or low weather variability. In addition, the used method for choosing the k clusters and the dimension of datasets is discussed.

4.2 Clustering Analysis

4.2.1 Choice of K clusters

To perform the clustering with K-means, a number of clusters k had to be chosen. The choice is the most difficult part, and the decision is the key factor of the algorithm. The number of clusters could be defined in several ways: empirically or based on the context and from the type of information which is looking for. In this context, both approaches were adopted.
The algorithm takes as input PVGIS dataset for Turin, with the range of the year

2005-2014, processed in two different ways, so that was possible to investigate more aspects of data:

- the first dataset took into account variables in Table 1.3 except for Day Of Year, Hour, and day. The mean over 24 hours was calculated, therefore, $\mathbf{x}_k \in \mathbb{R}^7$ consists in a vector (1×7). At the end, the dataset was composed of $(n \times 7)$, with $n = 3651$ considering the range of the year 2005-2014. Therefore, the objective was to extrapolate k clusters in which each cluster corresponds to a kind of day, on the basis of the weather. $k = 4$ was considered.

- the second one instead was used to study several patterns in a day. Therefore, $\mathbf{x}_k \in \mathbb{R}^{24}$ consists in a vector (1×24), since each entry has all the 24 hours of global solar irradiance. At end the dataset was composed of $(n \times 24)$, with $n = 3651$ as before. The desired result was to obtain k clusters in which each of them contained day with the same pattern production. To understand which number of clusters could have been suitable, the empirical method was performed.

The appropriate number of k clusters can be discovered in several ways and there is not a definitive approach. The one used in this context is called *Elbow Method*. The method consists of calculating and plotting the total WSS at varying the number of clusters. The total WSS, equation 4.2, sums up the WSS of each k cluster, giving a quantitative value of how much data are spread out.

$$WSS_{Total} = \sum_{c=1}^{K} \sum_{k=1}^{N} (\mathbf{x}_k - \bar{\mathbf{x}}_C)^2 \tag{4.2}$$

Figure 4.1 depicts the total WSS for a growing number of clusters. From $k = 1$ to $k = 300$ was considered. Total WSS decreases, asymptotically to the x-axis, at increasing the number of clusters. Indeed, the minimum value of WSS is reached when $k = N$, obviously. The elbow method advises taking a value in which the derivative is higher, indicating an abrupt change. Figure 4.2 shows values in which

the total WSS start to be more or less steady. In the end, $k = 4, 5, 6, 7$, and 8 were tested to examine if different results could have been obtained.

Figure 4.1: Total WSS for a growing number of clusters

4.2.2 Weather recognition

As explained before, two datasets were generated. Here will be illustrated the analysis on the first type, to exploit K-means to group day with the same weather. Since PVGIS dataset has hourly data, the mean of each variable was computed to get a representative value of the day. Consequently, the $\mathbf{x}_k$ observation consisted in a vector of mean values, in which $\mathbf{x}_k \in \mathbb{R}^7$, and 7 was number of the retained features. Since the available dataset contained hourly data from 2005 to 2014, many analysis could be performed, for instance, a matching between seasonality and clusters so that was possible to catch similar days, Indeed, some clusters appear only in a season and not in the others. Figures 4.4 and 4.3 show months with a low energy production, winter and autumn. Each figure shows all month's days depicted with a different colour, based on the cluster which belongs to.

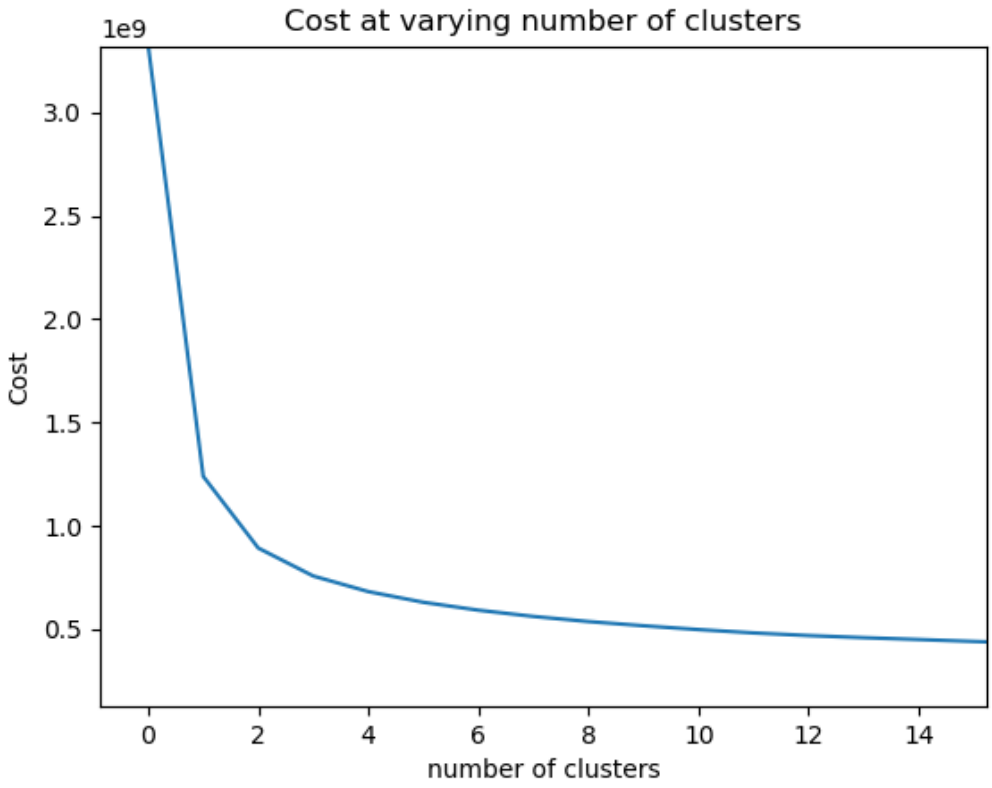
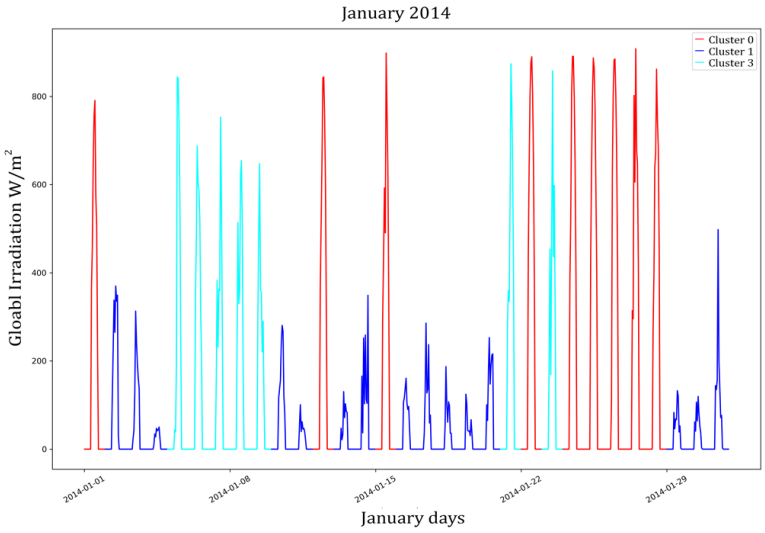
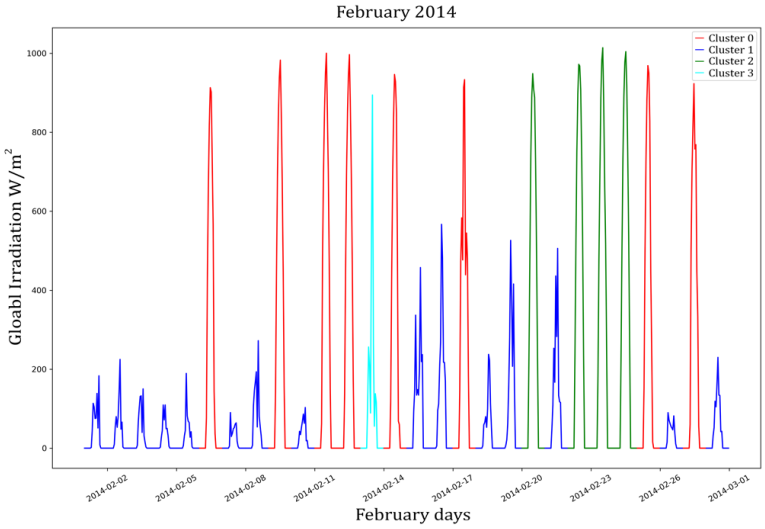


Figure 4.2: Zoom in the elbow curve for choosing k

First of all, there is a clear difference between some clusters on the basis of the global solar irradiance value. The blue cluster depicts a day in which the production is small and the day is highly variable, In addition, this cluster is always present in autumn and winter. On the contrary, the green cluster is not always present. Looking at the y-axis, and it is evident how this cluster contains days with a higher global solar irradiance value, so as a more normal behaviour than others day. This cluster is not always in all months, such as November, December and January, which are the colder months. This is a piece of helpful information if it is matched with the prediction of global solar irradiance illustrated above and several strategies could be established in energy management. Then, the other two clusters are in the middle.



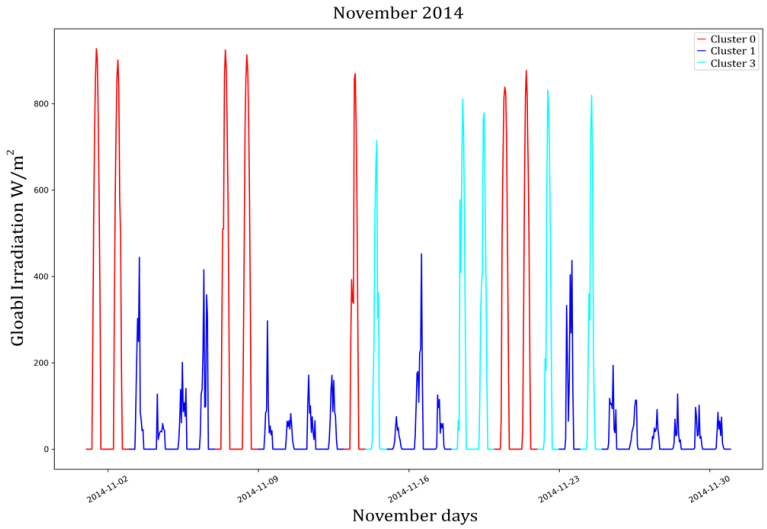
(a) January 2014



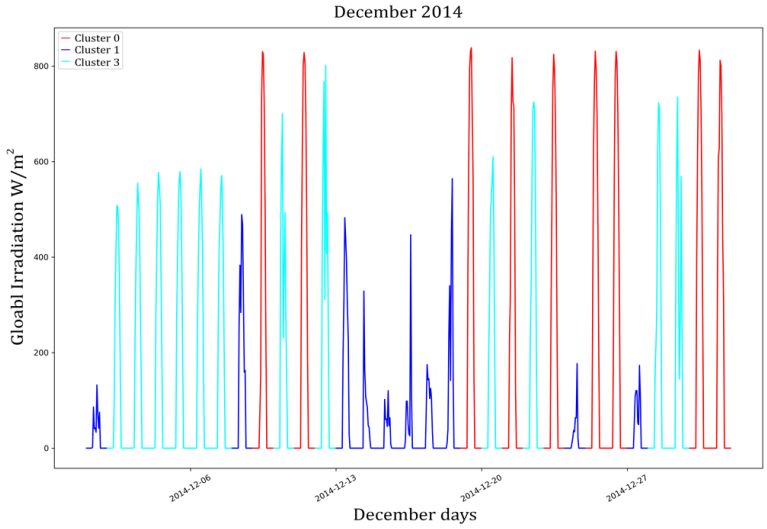
(b) February 2014

Figure 4.3: Month with lower energy production: January and February

The cyan cluster could indicate a higher global solar irradiance and little weather variation during the day; the red cluster could group those days in which there is a



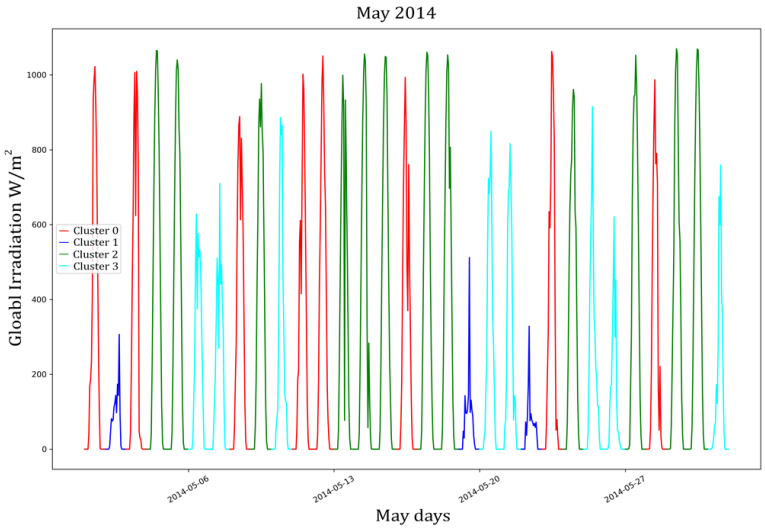
(a) November 2014



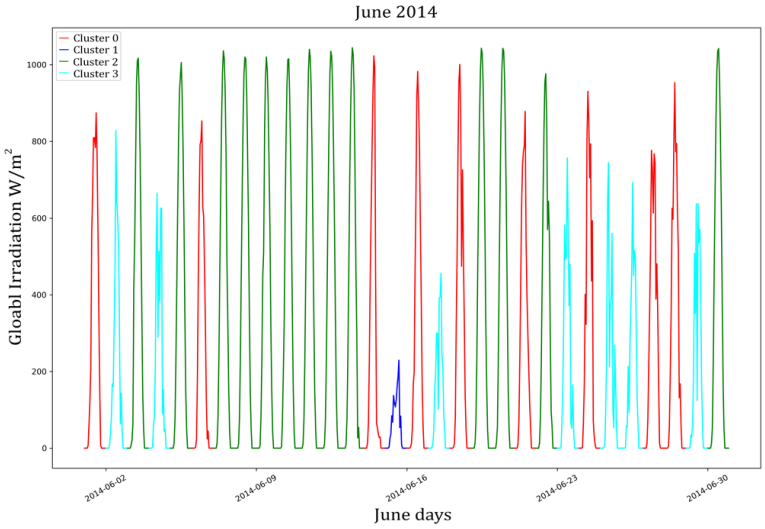
(b) December 2014

Figure 4.4: Month with lower energy production: November and December

high global solar irradiance but not as much the green cluster.



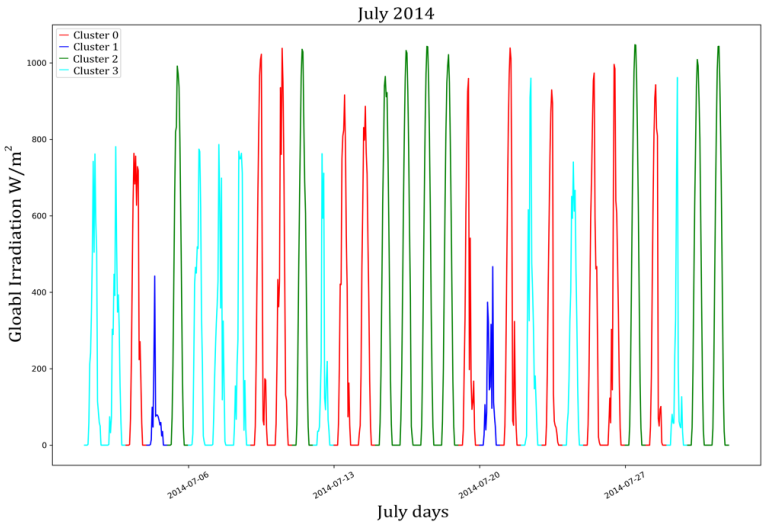
(a) May 2014



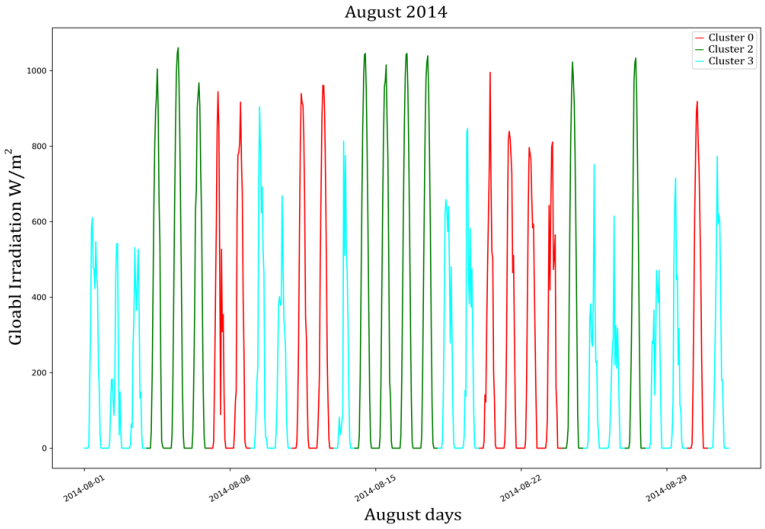
(b) June 2014

Figure 4.5: Month with medium and high energy production: May and June

Carrying on the analysis, Figures 4.5 and 4.6 show spring and summer months. The presence of the blue cluster is less frequent, indicating that few days have small



(a) July 2014



(b) August 2014

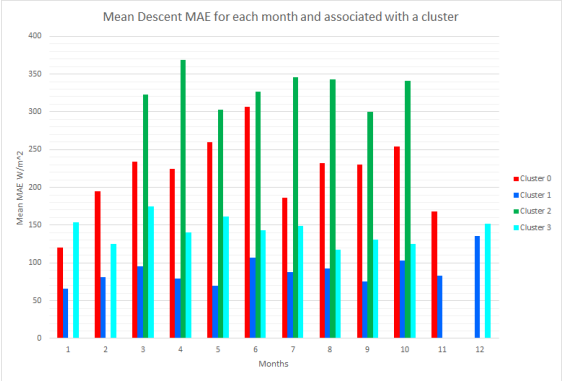
Figure 4.6: Month with medium and high energy production: July and August

global irradiance value and also few of them have a big daily variability. The green cluster is the one more frequent.

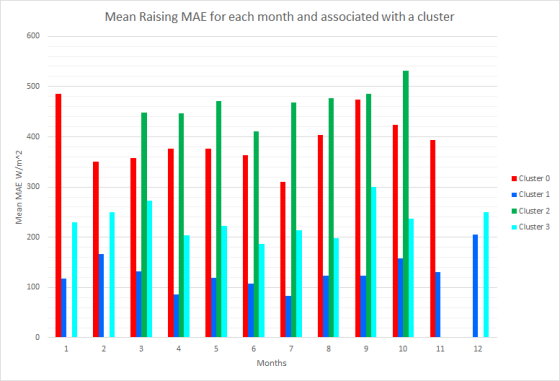
Summing up, K-means gives useful information about days by taking into account the values of global solar irradiance and the variability of the day. To understand if this information was useful to analyze the goodness of the prediction, a matching between prediction and clusters was performed. In order to do so, three other metrics were introduced: the error in raising, in descent, and the mean daily error defined as following:

- **raising MAE**: taking a single day, the metric calculated the difference between the real and the predicted value considering all the samples between the sunrise and the midday. In order to define the range of values to consider for each sunrise and midday, since both processes have a seasonal dependency, the elevation of the sun was the proxy, instead of using the hours;
- **descent MAE**: it takes into account values between the midday and the sunset, defining the range as the previous metrics. Besides, it does not consider the same hour of the midday, but just the hour after it, in order to avoid counting the same sample;
- **daily MAE**: it considers all the sample in 24 hours and calculated the MAE between the real and predicted values.

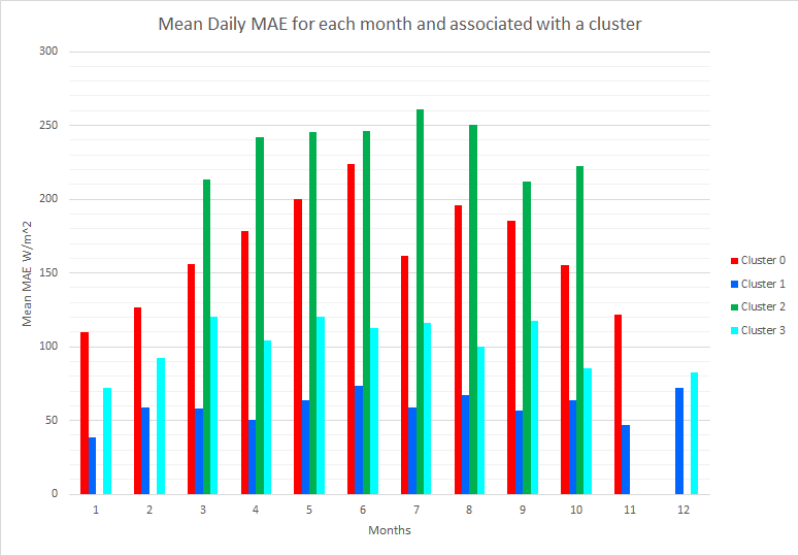
These errors were grouped for each cluster and each month to study the correlation between clusters and months as following: each type of error was calculated for a single day, averaged in each month, and then grouped for the associated cluster. The result is shown in Figure 4.7, in which in x-axis there is the month, while in the y-axis the Mean MAE. Bar plots 4.7a and 4.7b show the committed error from the best choice 3.1.4 and features in table 1.3 were used as input. Thanks to this data representation is interesting to see how months do not belong to all clusters, for instance January and February miss the green cluster - the one target as the cluster in which high global solar irradiance values are present. Also, November and December do not have all clusters. About the order of error, the blue cluster has a lower MAE than others, in both types of errors, raising and descending. On



(a) Mean Descent MAE



(b) Mean Raising MAE



(c) Mean Daily MAE

Figure 4.7: Mean monthly MAE, divided for raising, descent and daily

the contrary, the network is prone to make more mistakes in the red and the green cluster.

4.2.3 Pattern recognition

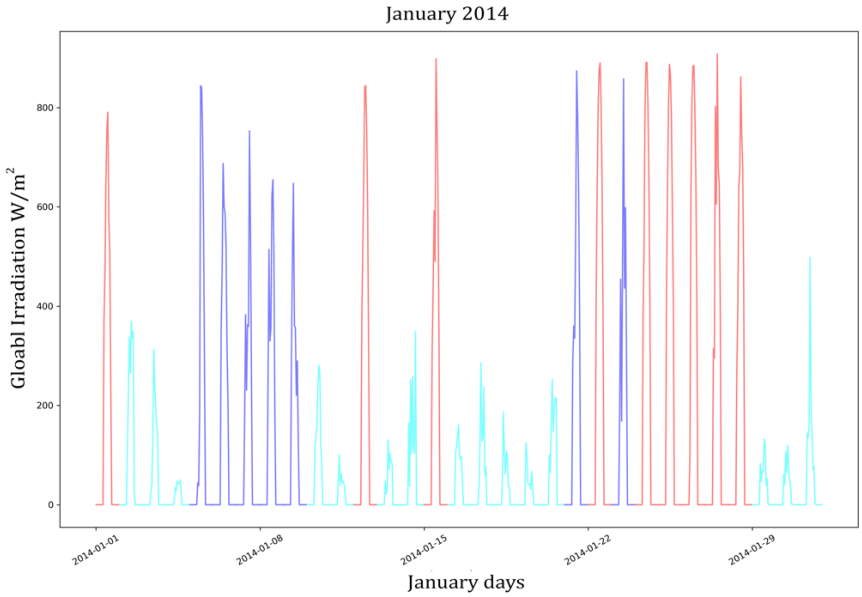
After having processed data for weather recognition, it was interesting to investigate if K-means could have allowed giving some helpful information about the daily global solar irradiance pattern. In order to do that, the PVGIS dataset was changed. The $\mathbf{x}_k \in \mathbb{R}^{24}$, that is each entry contained 24 values corresponding to the global solar irradiance at time t.
As said before, *Elbow method* suggested to compute the K-means algorithm with $k = 4, 5, 6, 7, 8$. Helpful result was obtained by looking to the clustering with $k = 7$, while for $k = 4$ results were similar to the previous approach. In fact, with a small value of k the same information of the previous approach was obtained.

Figures 4.8 and 4.9 depict the clustering with the new dataset and $k = 4$. Essentially, clusters are the same to the previous approach, therefore no more information could have been retrieved.

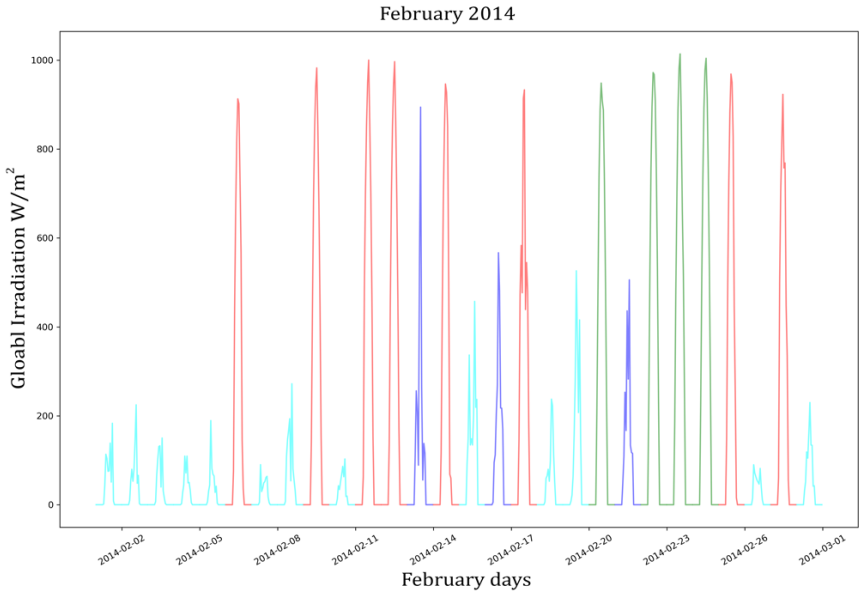
With a slightly higher number of clustering other characteristics were encountered. Analyzing $k = 7$ the order of global solar irradiance is not the only one which affects in the clustering process, but also the shape of the day, thanks to which is possible to create clusters with similar patterns. For instance, Figure 4.12 shows two types of days with a similar pattern. They are two different days taken from April 2016 (Figure 4.13a) and they can be classified as a day in which the evolution of global solar irradiance has high variability during the raising 4.13b and it is constant during the sunset. On the other hand, days shown in Figure 4.12b have high variability during the sunset. Another pattern can be seen during December days, aquamarine clusters groups days with the same shape (Figure 4.13).

K-means algorithm was the perfect algorithm for analyazing data on the basis of the thesis object. It gave helpful information about the type of day and also it allowed to introduce new metrics for evaluating the prediction, such as the committed error in raising and in descent These turn out very useful in context in which the prediction accuracy in a particular time of the day is crucial. In addition, the weather recognition improve the decision process in strategy management.

Another valuable information is regard the amount of k cluster which should be considered, since the higher is the number of cluster and slower is the total WSS. By contrast, a high number of clusters results intractable and the elbow method allows to proceed in the right, choosing a useful amount of clusters. Pattern recognition brings still other details about the daily global irradiance pattern. The knowledge of similarities between days can be considered for knowing in advance which could be the performance of a given neural network in some months, as seen in Figure 4.7.

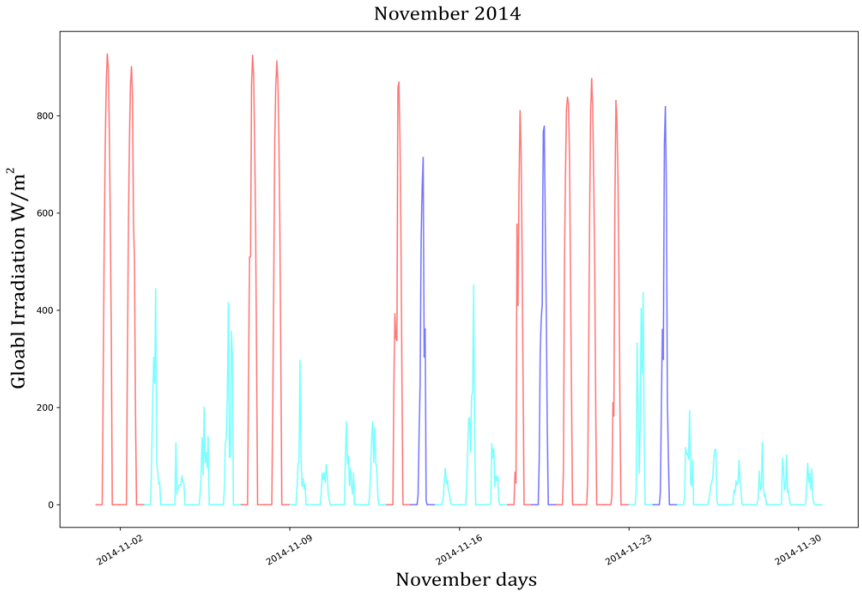


(a) January 2014

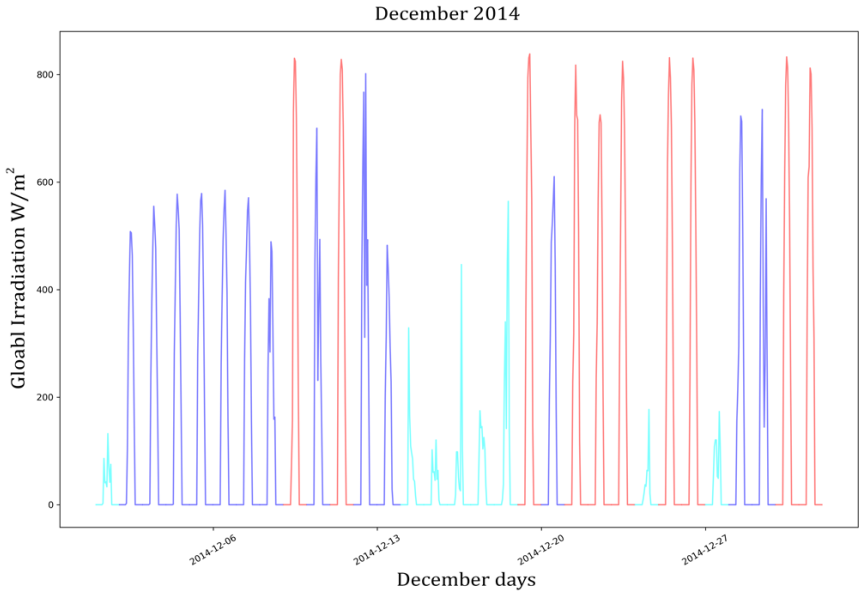


(b) February 2014

Figure 4.8: Month with lower energy production: January and February - Pattern recognition - $k = 4$

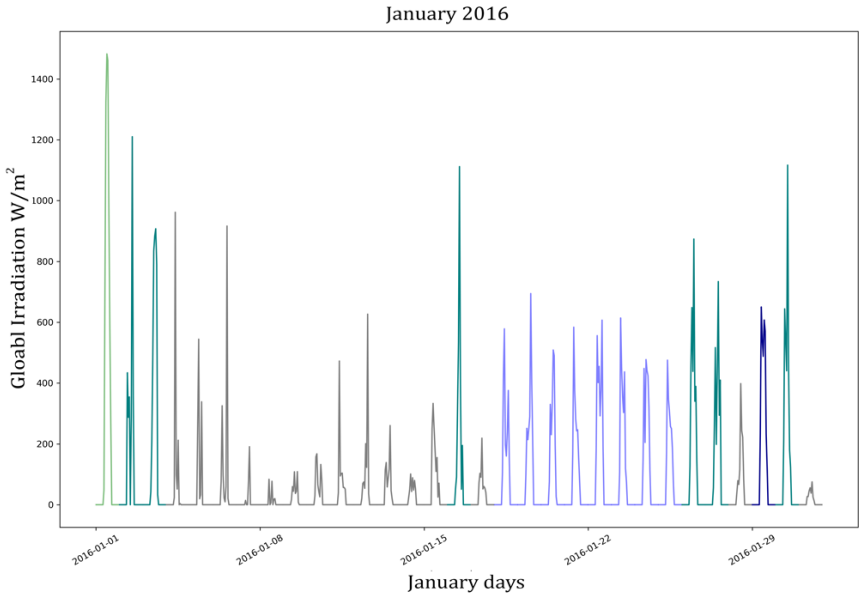


(a) November 2014

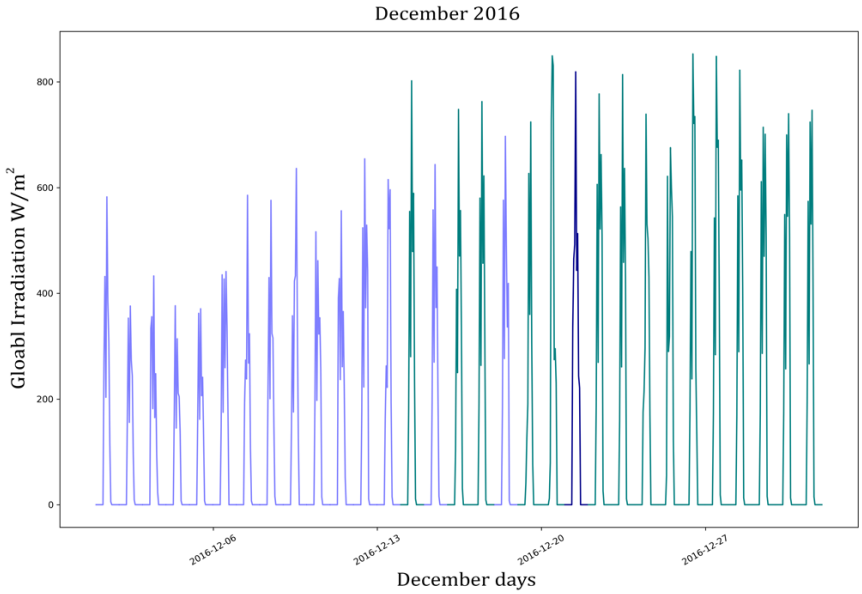


(b) December 2014

Figure 4.9: Month with lower energy production: November and December - Pattern recognition - $k = 4$

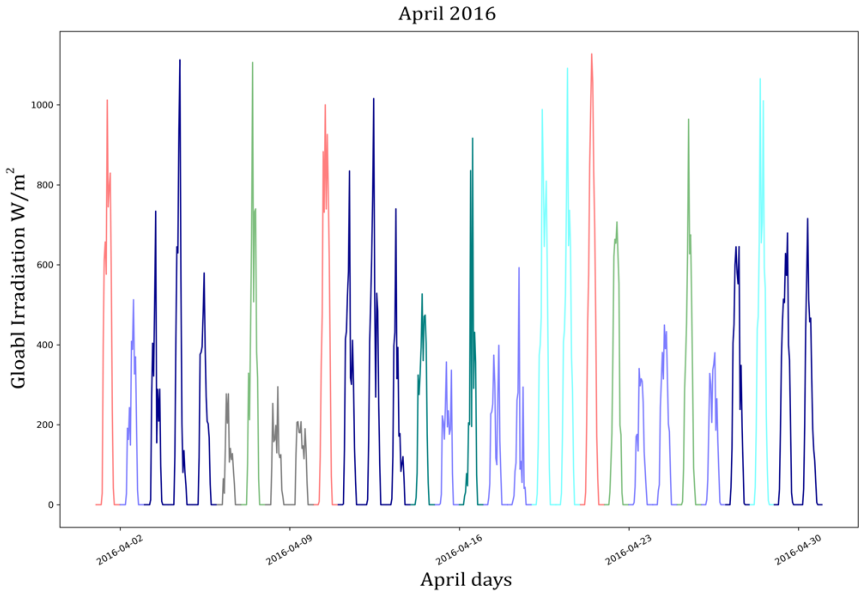


(a) January 2016

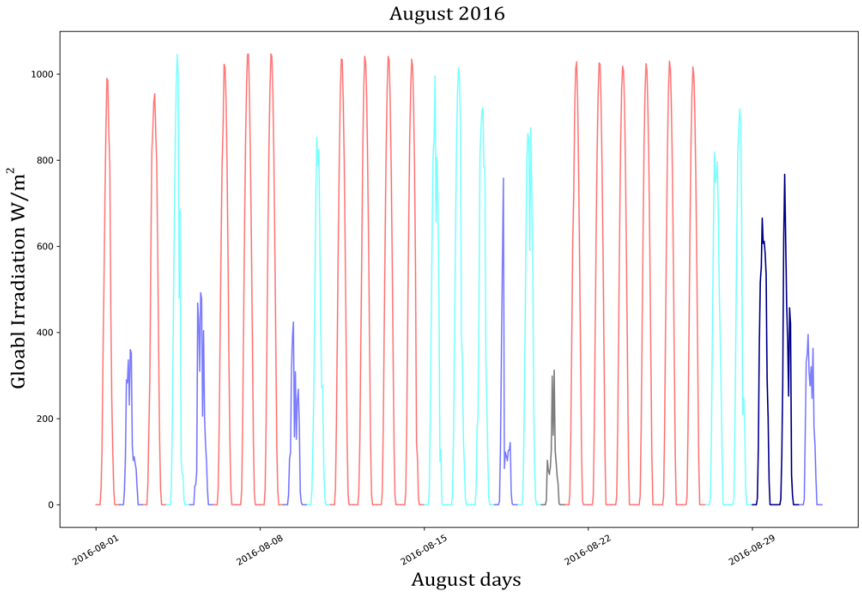


(b) December 2016

Figure 4.10: Month with lower energy production: January and December - Pattern recognition - $k = 7$

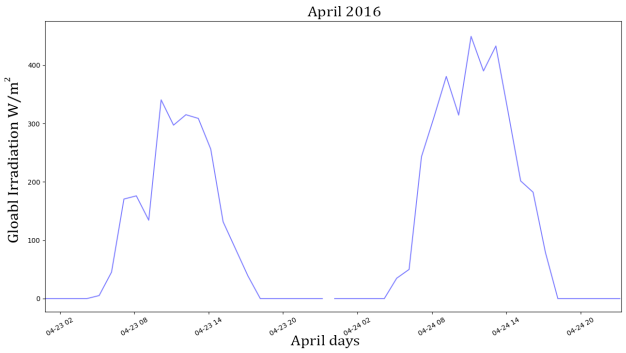


(a) April 2016

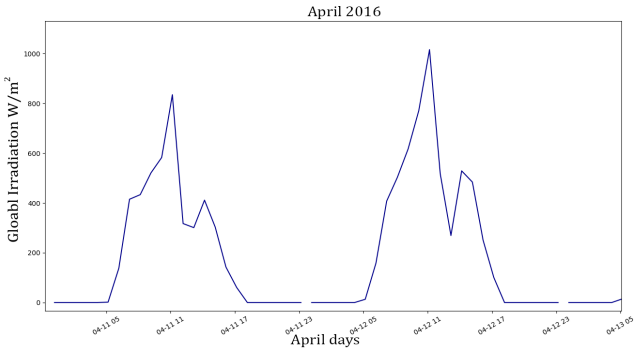


(b) August 2016

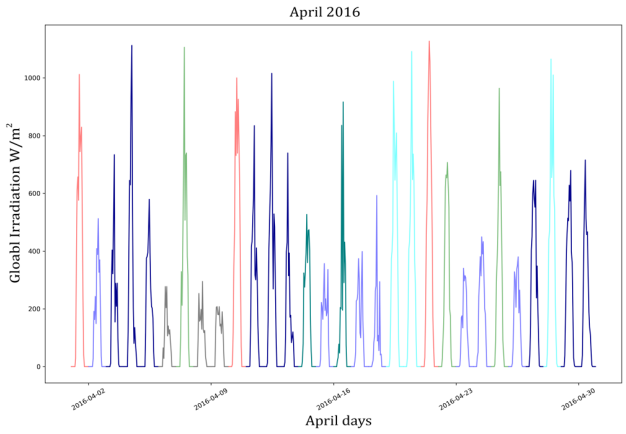
Figure 4.11: Month with lower energy production: April and August - Pattern recognition - $k = 7$



(a)

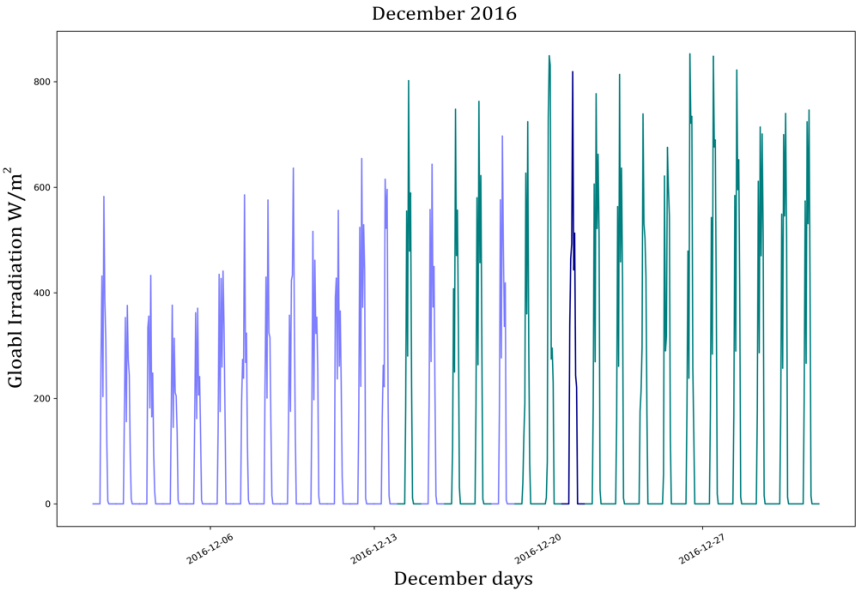


(b)

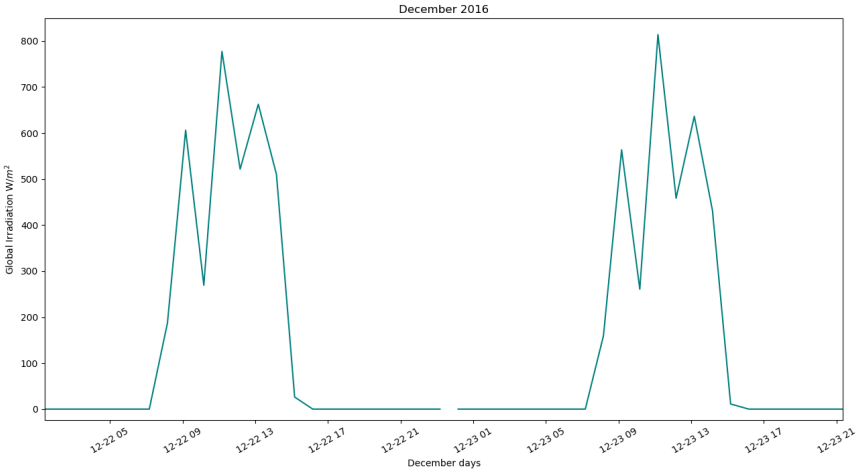


(c) April 2016

Figure 4.12: Comparison of two types of days with similar pattern in April



(a) December 2016



(b)

Figure 4.13: Comparison of two types of days with similar pattern in December

CHAPTER 5

Conclusions

The aim of the thesis wanted to give a new tool in a novel context, useful in several domains and mainly placed in energy management. The prediction of the renewable resources, by using machine learning technique, is a tool which can help the decision process in many contexts and use cases, and in order to do so, this tool should be affordable and reliable. The neural network techniques were considered and studied in different ways, by trying to understand if their properties would have been helpful in energy prediction. Indeed, a lot of difficulties were encountered, mainly due to their high dependency in the design, training process and, consequently, by the kind of fed data. Many tries of designing and training process were analysed for understanding how the neural networks behaved, such as the yearly, monthly, and seasonal approaches. The monthly approach was the one whose expected results were better than others. The comparison of all approaches depicted the yearly approach as the best one with lower errors and less prone to overfitting data. In fact, it is able to better generalise data, and it can be seen looking at the MSE error at increasing epochs.

Then, two types of neural networks were stressed in many ways, FNN and RNN. Literature shows that RNN performs better than FNN in time series analysis and in the prediction context. In this case, the differences between the networks are not so remarkable. In addition, FNN was able to perform even better than RNN.

Summing up, the analysis of these two techniques of machine learning brought new

aspects in energy prediction. The committed error from the NNs is higher than expected, and high dependency on the type of dataset was noticed. This is highly indicative about the kind of data should be used for obtaining good values of predictions, how they should be elaborated before data pre-processing and then training a network. The considered PVGIS and NSRDB tools give a good type of data, and many differences between them were analyzed from a different point of view: number of features, the type of features used for the neural network training, and the time granularity. With NSRDB data the neural network performed much better than data coming from PVGIS. The algorithm used for generating the dataset makes the difference and the network demonstrated very sensibly to it. Finally, the clustering analysis was used to extrapolate information and the obtained results were valuable and the comparison between the prediction and the real clustered day, which gave a sensible knowledge, remarking the difference between good and bad days, so as grouping type of day on the basis of the seasons. Indeed, the approach was a key point in the analysis, giving a lot of exploitable information in the decision making process in several domains (i.e. energy management).

In conclusion, the thesis is an introduction to this huge field in energy renewable prediction, in which a lot of work is required so as many improvements. Neural networks are a good tool and maybe exploiting some others techniques in deep learning could enhance the capability to learn and to predict global solar irradiance, with lower errors. This kind of tool can be applied in many fields of energy. For example, it can be used for energy management and cost reduction. Indeed, Neural Network allows to know which will be the amount of available energy in the next hour, and this information is helpful for deciding or managing the energy consumption in that hour. This can be translated in different ways on the base of context: for people in a residential building with a PV systems means to change their habits, accordingly with the amount of predicted available energy; for industries, the cost reduction can be very high, first of all, if they use renewable energy, and secondly the awareness of how much energy their PV systems produce can drastically change their financial statements. Indeed, the energy pricing strategy can be another context in which the application can be helpful. In this respect, the possibility to predict the availability

and the amount of energy allows to the industries to plan the quantity of energy they should buy from suppliers.
The tool results extremely useful for planning and managing the future available energy. Moreover, in this sense, it turns out extremely useful in critical situations or emergencies, just think of cities in which recurring black-out happens due to overload, weather condition, law restrictions for reducing the environmental impact of the power grid, and so on. The knowledge of the amount of energy in the next future can prevent critical situations, limiting the impact on both people and industries.
All these applications differ in requirements. Even though the presented Neural Network performs very well, some aspects can negatively affect on the base of context. For example, some applications require a smaller error during the day, other one needs more precision in predicting pick values, and so on. For these reasons, the tool needs to be improved in several aspects, starting from a higher computational capability and exploiting GPUs, which speed up the training phase and resulting faster 5 or 6 times than a CPU. It allows to study if increasing the number of hidden layers, the number of neurons, and so the number of parameters inside the Neural Network can distinctly improve performances. In parallel, the needs of more data, well structured or even previously classified in some way is preponderant in Deep Learning context.
At the end, machine learning techniques result extremely useful in these contexts, bringing to the next level many fields and laying the groundwork to sustainable thinking.

References

Bibliography

[1] Umair Shahzad. "The Need For Renewable Energy Sources". In: (Aug. 2015).

[2] R. Müller et al. "The CM-SAF operational scheme for the satellite based retrieval of solar surface irradiance—A LUT based eigenvector hybrid approach". In: *Remote Sens. Environ* (2009), pp. 1012–1024.

[3] R. Müller et al. "A new algorithm for the satellite-based retrieval of solar surface irradiance in spectral bands." In: *Remote Sens.* (2012), pp. 622–647.

[4] Diederik P. Kingma and Jimmy Lei Ba. "Adam: A method for stochastic optimization". In: *3rd International Conference for Learning Representations, San Diego* (2015).

[5] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. "An Empirical Exploration of Recurrent Network Architectures". In: Proceedings of Machine Learning Research 37 (July 2015). Ed. by Francis Bach and David Blei, pp. 2342–2350. URL: `http://proceedings.mlr.press/v37/jozefowicz15.html`.

[6] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *CoRR* abs/1502.03167 (Mar. 2015). URL: `http://arxiv.org/abs/1502.03167`.

Webography

[7] Photovoltaic Geographical Information System. *Overview of PVGIS data sources and calculation methods*. URL: http://re.jrc.ec.europa.eu/pvg_static/methods.html.

[8] National Renewable Energy Laboratory (NREL). *Solar Resource Glossary*. URL: https://www.nrel.gov/grid/solar-resource/solar-glossary.html.

[9] National Renewable Energy Laboratory (NREL). *U.S. Data - National Solar Radiation Database (NSRDB)*. URL: https://nsrdb.nrel.gov/current-version.

[10] National Renewable Energy Laboratory (NREL). *International Data - National Solar Radiation Database (NSRDB)*. URL: https://nsrdb.nrel.gov/international-datasets.

[11] National Renewable Energy Laboratory (NREL). *Physical Solar Model - National Solar Radiation Database (NSRDB)*. URL: https://nsrdb.nrel.gov/current-version#psm.

[12] Michael Nielsen. *Neural Networks and Deep Learning*. June 2019. URL: http://neuralnetworksanddeeplearning.com.