Universidad Nacional Autónoma de México

Instituto de Energías Renovables

Instituto de Ingeniería

Escuela Nacional de Estudios Superiores-Juriquilla

Meteorological data imputation and its impact on EnergyPlus simulations

T E S I S

para obtener el título de:
Ingeniero en Energías Renovables

presenta:
Alejandro Ismael Guadarrama Zendejas

Director de Tesis:

Dr. Guillermo Barrios del Valle

Temixco, Morelos a 02 de Septiembre de 2022

Acknowledges

I want to thank my parents Carolina and Ismael for bringing me the support and advice throughout my lifetime. To my sister for showing me to not give up from achieving your dreams.

I want to thank UNAM for giving me the opportunity for being part of this great community. To the Renewable Energies Institute for giving me the knowledge and opportunities for growing professionally.

Special thanks to my advisor Guillermo Barrios del Valle for his patience and guidance and for providing the tools and support that I needed to complete this work. To José de Jesús Quiñones Aguilar for his work on the meteorological station ESOLMET-IER from which the data was obtained since without it, this work wouldn't be possible. To Kevin Alquicira for his maintenance and access to the computational resources needed for the development of this work.

To all IER-UNAM community. To the academics and professors that guided me in this journey. To Ramón Tovar Olvera for introducing me into the investigation field in the buildings energy group. To Maribel, Carlos, Magali, Claudia and Miguel for your support and help when I needed it the most.

To my roomies Zyanya and Mariana for providing me your knowledge and wise experiences and for giving me the company that I didn't know I needed. To Augusto, Emilio, Jesús, Medel, Roy, Jorgito, Benita, Demián and Taisha for your valuable friendship, knowledge and support.

To all the 6G members that taught me a new way to view life, for encouraging me to rise above my limits when things get hard and for being the group of people that made me feel comfortable with being myself.

Abstract

Missing data is a common problem when managing big amounts of data. And meteorological data, obtained from meteorological stations is not the exception. Imputation is defined as the methods to fill missing data with plausible values [1], these methods can help to solve this problem to have reliable data to be processed and used for further investigations.

The methodology is divided in two parts; training and comparison of imputation models, and the effect of imputation in an EnergyPlus simulation. The first part presents the methodology of training, validation and testing of a hybrid convolutional neural network long short term memory model (CNN-LSTM). The methodology to determine the orders and the testing of a seasonal auto regressive integrated moving average (SARIMA) is also presented. Both of these models are used to impute the day ahead of global horizontal irradiance (I_g) measurements. Also a comparison between both models is done, where by taking into account the value of the metrics is determined that the CNN-LSTM model outgrades SARIMA model.

The second part presents the impact of an imputation on thermal behavior of a building simulated on EnergyPlus. For this purpose the I_g measurements of some selected days in a year were removed, then the CNN-LSTM model is used to impute these selected days. By using other metrics the impact of these imputations on the zone mean air dry bulb temperature (T_i) on the thermal zones of the building is evaluated. Also this part presents an analysis of the factors that contribute to increase the impact of imputations on T_i.

Resumen

Los datos faltantes son un problema común cuando se manejan grandes cantidades de datos. Y los datos de clima, obtenidos de estaciones meteorológicas, no son la excepción. La imputación se define como los métodos para completar datos faltantes con valores estimados plausibles [1], estos métodos pueden ayudar a solucionar el problema de datos faltantes y así obtener datos confiables para ser procesados y usados en investigaciones futuras.

La metodología se divide en dos partes; el entrenamiento y comparación de modelos de imputación, y el efecto de imputación en una simulación en EnergyPlus. En la primera parte se presenta la metodología para entrenar, validar y probar una red neuronal híbrida convolucional long-short term memory (CNN-LSTM por sus siglas en inglés). También se presenta la metodología para la determinación de órdenes e implementación de un modelo estacional auto regresivo integrado de media móvil (SARIMA por sus siglas en inglés). Ambos modelos se usan para imputar un día completo de irradiancia global horizontal (I_g). Así también, se hizo una comparación entre estos dos modelos, donde por medio de los valores de las métricas se determinó que el modelo CNN-LSTM superó a el modelo SARIMA.

En la segunda parte se evalúo el efecto de la imputación en el comportamiento térmico de un edificio simulado en EnergyPlus. Para este propósito se removieron los datos de la I_g en ciertos días seleccionados del año, después se utilizó el modelo CNN-LSTM de la primera parte del trabajo para imputar estos días seleccionados. Utilizando otras métricas se evalúo el impacto de estas imputaciones en la temperatura interna (T_i) de diferentes zonas térmicas del edificio. También se presenta un análisis de los factores que pueden influir en que las imputaciones tengan un mayor efecto sobre la T_i.

Contents

List of Figures

List of Tables

List of Symbols

Acronyms

ACF Autocorrelation Function

ANN Artificial Neural Network

AR Autoregressive Model

ARIMA Autoregressive Integrated Moving Average

ARMA Autoregressive Moving Average

BPTT Backpropagation Through Time

BS Base Simulation

BS Batch Size

CNN Convolutional Neural Network

DL Deep Learning

ESOLMET-IER Solarimetric and Meteorological Station

FFANN Feed Forward Artificial Neural Network

FFL Feed Forward Layers

GRU Gated Recurrent Unit

IS Imputed Simulation

LSTM Long Short Term Memory Neural Network

MA Moving Average Model

MIMO Multiple Input Multiple Output

ML Machine Learning

MLP Multi Layer Perceptron

MSE Mean Square Error

PACF Partial Autocorrelation Function

PV Photo Voltaic

RNN Recurrent Neural Network

SAR Seasonal Autoregressive

SARIMA Seasonal Autoregressive Integrated Moving Average

SMA Seasonal Moving Average

TMY Typical Meteorological Year

Variables

η Learning Rate

ρ_k ACF Coefficient

b Intercept

C Cost Function

C_t Recurrent State

$E(t)$ Time Series Error o Irregular Component

E_d Daily Solar Energy $[\frac{Wh}{m^2}]$

E_d^p Predicted Solar Energy $[\frac{Wh}{m^2}]$

E_d^{err} Daily Energy Absolute Error $[\frac{Wh}{m^2}]$

f_t Forget Gate

h_t Recurrent Neural Network Output Signal

hi_t Hidden State

I_g^p	Predicted Global Horizontal Irradiance $[\frac{W}{m^2}]$
I_g^{err}	Irradiance Absolute Error $[\frac{W}{m^2}]$
i_t	Input Gate
m	Slope
max_t	Maximum Error Time [days hh:mm]
o_t	Output Gate
R^2	Correlation Coefficient
rh	Relative Humidity [%]
T_i^p	Zone Mean Air Temperature reported by Imputed Simulation $[^oC]$
T_i^{err}	Temperature Absolute Error $[^oC]$
T_o	Air Dry Bulb Temperature $[^oC]$
T_{err}^{max}	Maximum Temperature Absolute Error $[^oC]$
$tanh$	Hyperbolic tangent
V_t	Vanishing time [days hh:mm]
W	Weights Matrix
γ_k	Covariance
σ	Sigmoid function
$\tilde{C}_t$	Candidate Recurrent State
$C(t)$	Time Series Cyclical Component
D	Seasonal Differentiation Order
d	Differentiation Order
$E_d^{\%err}$	Daily Energy Percentage Error [%]
f	Activation function
h	Node Output Signal

h_c	Output of Convolution
I_b	Direct Normal Irradiance $[\frac{W}{m^2}]$
I_d	Diffuse Solar Irradiance $[\frac{W}{m^2}]$
I_g	Global Horizontal Irradiance $[\frac{W}{m^2}]$
p	Autoregressive Order
P	Seasonal Autoregressive Order
q	Moving Average Order
Q	Seasonal Moving Average Order
$ReLu$	Rectified Linear Unit
$S(t)$	Time Series Seasonal Componet
$T(t)$	Time Series Trend Component
T_i	Zone Mean Air Dry Bulb Temperature $[^oC]$
T_{osc}	Yearly Mean Zone Mean Air Temperature Oscilation $[^oC]$
z	Weighted Sum

Introduction

Weather data can be useful for multiple applications such as; weather forecasting, evaluation of solar, wind or rain sources for renewable energy investing. For the case presented in this thesis it is used for energy building simulations. However the recording of data in weather stations and measurement devices tend to fail for several reasons, such as energy shutdowns, sensor malfunction, maintenance, etc. Therefore missing data is one of the biggest problems when working with data sets.

Missing data can have consequences on the capacity to simulate or validate building simulations, which leads to a sacrifice in accuracy on the simulation, or to the repetition of experimental campaigns to ensure a validated simulation. This can waste time and economic resources. An imputation is defined as the process to fill missing data with plausible values [1]. Several imputation methods are used to solve this problem.

The objective of this work is to build a model to impute the day ahead measurements of global horizontal irradiance (I_g) using a hybrid convolutional neural network long-short term memory artificial neural network (CNN-LSTM) and compare its performance with a seasonal autoregressive moving average model (SARIMA). Another objective is to study the impact of the imputation of I_g on the zone mean air dry bulb temperature (T_i) of the thermal zones of a building simulated on EnergyPlus in comparison with the T_i obtained when the building is simulated with the measured I_g.

Chapter 1 is the literature review, which presents the history of the artificial neural networks (ANN) and the models used for predicting and imputing weather data. Chapter 2 presents the theoretical background of ANN and SARIMA model. Chapter 3 presents the methodology to select the best I_g imputation model comparing a trained CNN-LSTM and a SARIMA model. Chapter 4 shows the impact of the CNN-LSTM imputed I_g on the interior temperature of the building thermal zones. And Chapter 5 presents the conclusions of this work.

Chapter 1

Literature review

This chapter presents the literature review about the history of artificial neural networks (ANN). It also presents some of the models that are used to forecast or to impute weather data, as well as the advantages that imputation and forecasting of weather data can have.

Section 1.1 presents a brief history of ANN's and some of its major breakthroughs. Section 1.2 presents the models used to predict or impute weather and mainly irradiance data, to get an insight on how are usually applied and typified.

1.1 History of ANN´s

The beginnings of ANN's can be referred to 1943, when McCulloch and Pitts [2], one a psychologist and the other a mathematician, built a theoretical model based on the function of neurons. This was done considering the neuron as a functional logic device. Frank Rosenblatt [3] proposed the Perceptron in 1958 which was a linear discriminator model based on the McCulloch and Pitts model and that preceded the basic principle of the current modern neural networks structure. In 1969 Minsky and Papert [4] on its book 'Perceptrons, An Introduction to Computational Geometry', published the limitations that the Perceptron had. It was pointed out that perceptrons could not solve the classification problem of two types of linear inseparable samples. In 1984 Hinton and Sejnowsky [5] proposed the first multi-layer network learning algorithm known as the Boltzmann machine model. In 1984 Hopfield [6] proposed to change one of the main parts of the neurons, the activation function, from discrete to continuous.

In 1986 based on Boltzmann machine algorithm D.E. Rumelhart et al. [7]

proposed the backpropagation algorithm to solve the weights correction of the multi-layer neural network. Which enabled these models to solve a larger variety of practical problems than before. This increased the attention to the development of different neural networks variants. One of these variants were the recurrent neural networks (RNN), first proposed by Robinson and Fallside [8] in 1987 in which part of the output of the network feedbacks to the input to create some sort of memory storage. In 1997 Hochreiter and Schmidhuber [9] introduced the Long Short-Term Memory neural network (LSTM). Which is a kind of RNN with multiplicative gate units that allows the model to learn faster than the traditional RNN. In 2006 Hinton et al. [10] proposed Deep Learning (DL) as a new field of machine learning (ML) which is defined as an ANN architecture model with multiple hidden layers and the model is trained through large scale data training.

1.2 Models for weather imputation

The presented models were on its majority used for forecasting rather than for imputation. Imputation is defined as the process to fill missing data with plausible values [1]. While forecasting or prediction can be defined as the process to tell future time series data given the past values of itself or other variables. This implies that predicted data can also be used to impute time series data. For this reason the words to predict and to impute can be used interchangeably.

Solar irradiance forecasting is used in order to anticipate the changes on power output in photo voltaic (PV) plants. Solar power ouptut has a high variability due to several weather variables, for example cloudiness, temperature, wind speed and solar irradiance itself. This variability causes issues on the electric grid, such as voltage surges, reverse power flows, variations in frequency harmonic distortion in current and voltage waveforms etc [11]. Therefore a reliable PV output forecast would decrease uncertainty, enhancing stability and improving economic viability. On the other hand for building energy consumption, high quality data has become important to study how to improve energy efficiency [12]. Missing data can affect the research outcomes. Therefore a reliable data imputation can make more viable to make an analysis and identification of energy consumption patterns, for data driven decisions.

Prediction models for solar irradiance can be classified by the time horizon. Which is the future time period for the output prediction or the time duration between actual and effective time of prediction [11]. Time horizons

are divided by: very short term, from a few seconds to one hour, short term, from one hour to 6 hours, medium term, from six hours to twenty four hours, and long term, for more than twenty four hours. However there is no widely agreed classification criterion [13].

Imputation can be classified by the frequency and extension of the missing data. Where it is divided as random missing and continuous missing data [12]. In the first case, the missing data happens with a high and irregular frequency but for short periods of time. Meanwhile the continuous missing data happens in long extensions of time. This work will be focused in solving continuous missing data of weather, mainly on the global horizontal irradiance.

A multi-step prediction model is necessary in order to impute continuous missing data. For multi-step ahead prediction, different modeling approaches can be applied; iterated, direct and multiple input multiple output (MIMO) approach [14]. For an iterated approach a model is trained to give a 1 step ahead prediction, this prediction is then used as an input to give the next step prediction. This process is iterated until reaching the desired time horizon [15]. A direct approach consists on constructing the same amount of models as the time steps that needs to be predicted. This approach takes a significantly larger amount of computational time for a large number of time steps [14]. MIMO approach consists on constructing a model able to predict the needed amount of time steps at once, usually feeding the model with a multi-step input. The MIMO approach has shown to be more accurate to predict hourly solar irradiance for up to 1 to 12 hours ahead than the iterated approach, and needs less computational time since it requires to train a single model unlike the direct approach [15].

Weather prediction models can be divided into three kinds of models, which are physical, statistical and machine learning (ML) models [11]. The most studied have been statistical and ML models [16], which are the ones used on this work. Physical models use external variables, such as other weather and geographical variables. An example of this kind of model is the numerical weather prediction which uses regional or even global atmospheric features to forecast up to 15 days ahead. For this approach it is needed a larger amount of data and resources than statistical or ML models [11]. Statistical models improves the prediction accuracy by minimizing a loss function with a linear process. Usually statistical models inputs are endogenous, which means that these models use only the historical data of the predicted variable [17]. Statistical models need less computational power than ML or physical models [16]. Meanwhile ML models optimize their loss function with a non linear process and typically use an exogenous

input, which means that these models use the historical data of the predicted variable as well as other external weather variables [17].

Statistical models are one of the most traditional time series prediction methods. One of these models, the auto regressive integrated moving average (ARIMA) model has the best prediction accuracy [16]. In the case of ML models, feed forward artificial neural networks (FFANN), also called in some instances multi layer perceptron (MLP) has an equivalent performance in comparison with ARIMA models for solar irradiance prediction in certain variability conditions and time horizons [18]. Nevertheless other ANN variants such as RNNS like LSTM neural networks have a better accuracy for predicting solar irradiance than FFANN models [19]. However the LSTM model takes more computational resources than FFANN [15]. Other RNNs variants like the gated recurrent unit (GRU) has a similar performance in comparison with the LSTM [20].

Other novel variants have proven to have a better accuracy than simple FFANN or LSTM models. One of these is a bidirectional LSTM imputation method, used for energy building consumption data [12]. Another example is a hybrid ARIMA-LSTM model, used for daily global horizontal irradiance [21]. Lastly a hybrid convolutional neural network with LSTM, used for PV power forecasting [22].

It is important to acknowledge that the models presented on the last paragraph are the ones that have had the best performance just in the specific cases exposed. This means that in other circumstances, other models could have a better performance than them, depending on the variability of the data, frequency of the measured data, time horizon, etc. This makes hard to affirm which model has the best general performance, since the mentioned variables change with each study case addressed [16].

Chapter 2

Models for weather imputation theoretical background

This chapter presents a conceptual introduction of two kinds of models. First the artificial neural networks (ANN) which are based on machine learning (ML) models, and second, a seasonal auto regressive integrated moving average (SARIMA), which is a statistical model. Section 2.1 presents the way ML models are classified by its way of learning, the basic functioning of ANN's and multi layer perceptrons (MLP), also how they are trained and other types of ANN's such as recurrent neural networks (RNN) and convolutional neural networks (CNN). Section 2.2 presents the statistical models used in this work, some assumptions of these models, and the equations that describes auto regressive moving average (ARMA) models, non seasonal auto regressive moving average (ARIMA) and SARIMA models, as well as a way to determine the parameter orders of these models.

2.1 ANN models

On traditional ML models, relevant features of the data needs to be extracted so that the model learns from it [23]. However for the deep learning (DL) models, which includes all kinds of multi layer ANN architectures, these features are learned automatically.

ML, and so DL models can be classified by its way of learning; supervised, unsupervised and reinforcement learning [24]. In supervised learning, labeled data is delivered to the model in order to be trained. This means

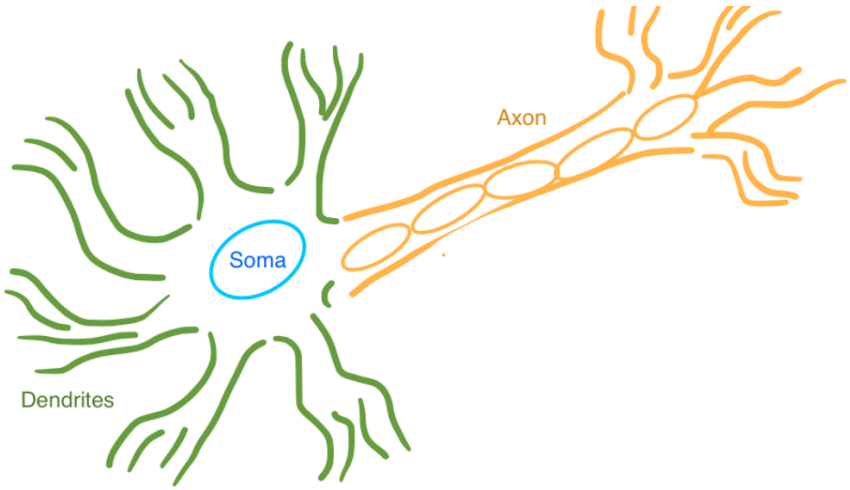


Figure 2.1: Organic neuron diagram.

that the training data comes in x and y pairs, where the objective of the model is to predict certain output (y') given any new input (x'). This approach is used to solve classification or regression problems. Unsupervised learning is fed with unlabeled data. This kind of learning can solve clustering problems which consists on finding a partition on the observed data without having explicit labels indicating a desired partition. Reinforcement learning is considered a combination between supervised an unsupervised learning. Instead of feeding the pairs of data with the correct desired outputs, as on supervised learning, is fed with an indication if the done action after given an input is correct or incorrect. If an action is incorrect, the problem of finding the correct action remains. This kind of learning is used in control strategy problems.

2.1.1 An inspiration on nature

As it was mentioned on the last chapter, ANN models are based on a simplified way that organic neural networks work on the brain [25]. Organic neural networks are composed of cells called neurons, the neuron components, as seen on Figure 2.1, can be simplified in three parts; the soma, in blue color, is the body of the cell, dendrites, in greeen color, are the input ramifications and the axon, in orange color has the output ramifications. Meanwhile on Figure 2.2 is presented a diagram of neuron synapses, which are the spaces that are between dentrites and axon terminals. The neuron receives signals

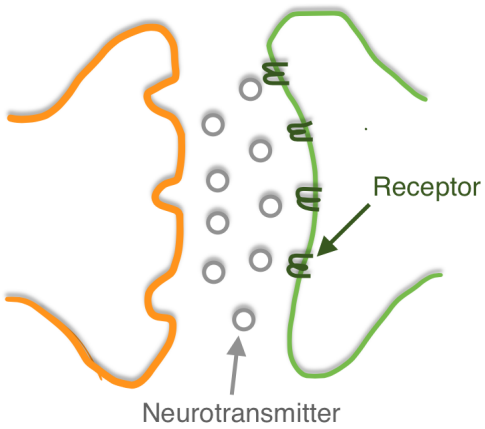


Figure 2.2: Neuron synapses diagram.

from other neurons through the terminals of its dendrites. These signals are summed up in the soma and if this sum surpasses certain electrical threshold the neuron activates, transmitting an output signal through its axon until reaching the axon terminals. This output signal is then passed to the next layer of neurons by neurotransmitters, that passes through synapses to the receptors of dendrite terminals of the next layer of neurons.

The neurons communicate with each other through electrical and chemical signals. The impulse that travels in the neuron and through the axon is electrical, while the signal transmitted through synapses are chemical. This chemical signal is regulated by neurotransmitters which work by inhibiting or reinforcing the signals on the synapses of each neuron depending on the use or disuse of the neural pathways [26]. This is called neuroplasticity and allows the brain to be adaptable to changes and to learn new things through all of our lifetime.

Because ANNs are a mathematical model, signals are numerical. The neuron or node is the basic unit that composes the ANN and can be explained by the next equations.

$$z = \sum_{i=1}^{n} x_i w_i, \tag{2.1}$$

where x is a n length array of the input signals and w is a n length array of the weights. Therefore z is the weighted sum of the node inputs. The other

equation is,

$$h = f(z), \tag{2.2}$$

where h is the output signal and f is the activation function.

Activation function is a non linear function that is applied to the sum of the weighted signals received by each node. This non linearity gives ANNs the ability to be stacked into several neurons layers so that it can fit the given data [27]. If this function were lineal, there would be no purpose on stacking nodes together since this would only be equal to a single equivalent linear regression. Even if there exists a wide variety of activation functions, the ones used in this work are the sigmoid, hyperbolic tangent and rectified linear unit ($ReLu$) functions. Sigmoid function or logistic function is described by:

$$\sigma(z) = \frac{1}{(1 + e^{-z})}, \tag{2.3}$$

where z represents the sum of the weighted inputs. The hyperbolic tangent function is described by:

$$tanh(z) = 2\sigma(2z) - 1, \tag{2.4}$$

And ReLu activation function is described by:

$$ReLu(z) = max(0, z), \tag{2.5}$$

As shown on Figure 2.3 sigmoid and hyperbolic tangent functions are 'S' shaped. However the sigmoid function, on Figure 2.3(a) has output values in a range from 0 to 1. While hyperbolic tangent function, on Figure 2.3(b) has output values in a range from -1 to 1. While on ReLu, shown on Figure 2.3(c) when z has a value that is less than zero, the function outputs a zero, but when is bigger than zero it outputs the z value.

On Figure 2.4 is shown a representation of how information flows through the node of a perceptron where the inputs (x_1, x_2, x_3) are multiplied by its correspondent weights (w_1, w_2, w_3). These multiplications are summed (z) and then passed through a non linear activation function ($f(z)$) to give an ouput (h).

ANNs are conformed by perceptrons interconnected with each other. In a multi layer perceptron (MLP), neurons are connected densely which means every node's output from one layer becomes a part of the input of the nodes in the next layer. Figure 2.5 shows the architecture of a MLP. A MLP is conformed by three kinds of layers, input layers, hidden layers, and

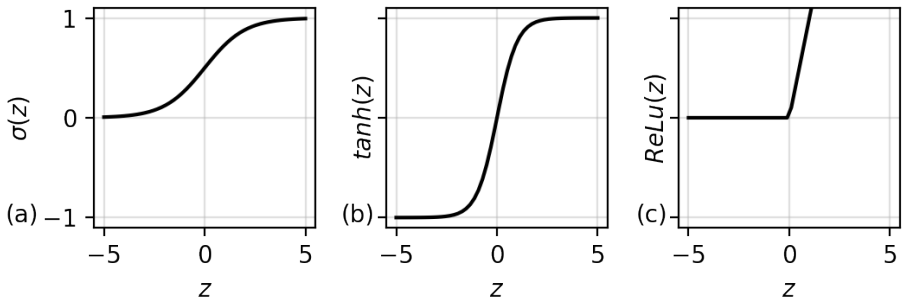


Figure 2.3: Graph of behavior of (a)sigmoid, (b)tanh and (c) ReLu activation functions.

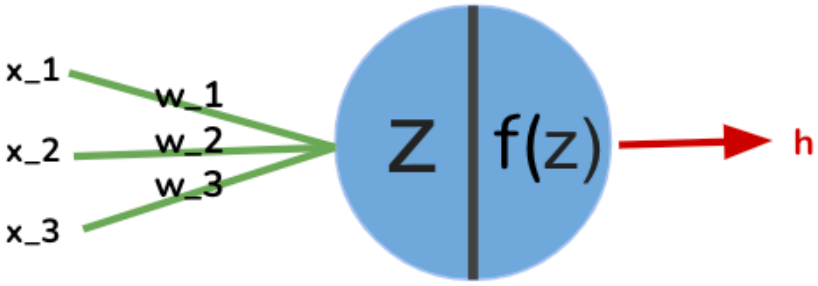


Figure 2.4: Graphic representation of a perceptron.

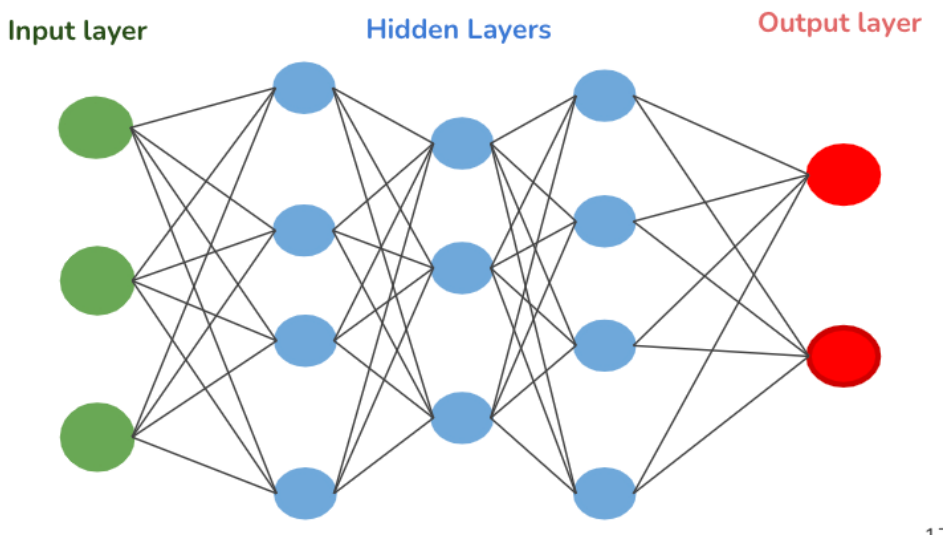


Figure 2.5: MLP graph representation.

output layers. Input layer is the one where the data goes into the network. This layer does not make any transformation to the data. On hidden layers information propagates and gets transformed by the activation function of each node until reaching the output layer. So with every additional layer in the MLP architecture, the information passes through another activation function of a weighted sum.

2.1.2 Gradient descent and backpropagation

The last section describes how the input data that feeds the ANN gets transformed by it. This section explains the algorithm that is used to train the ANN. Training is the process that is applied to the ANN so that the inputs produce the desired outputs [25]. For this, is necessary to introduce the concepts of the cost function, gradient descent and backpropagation.

As it was mentioned, supervised learning models needs to be trained with labeled data so that it can learn from it. This means that, in the training process, the ANN should have a desired output (y) to compare with the actual output of the ANN (h) [28]. This comparison is achieved with the cost function,

$$C(W) = \frac{1}{2n} \sum_{i=1}^{n} |y_i - h_i|^2, \tag{2.6}$$

where n is the number of samples and W is the weights matrix of the ANN architecture. This cost function is also called mean squared error (MSE). The cost function is an indicator of the performance of the ANN, where the main objective would be to bring $C(W) \approx 0$ or as small as possible by adjusting W.

Gradient descent is the algorithm of numerical optimization that is used to train several ML models. The gradient should be viewed as the vector of maximum change rate in any function. Gradient descent works by subtracting the gradient iteratively to the parameters in order to minimize the error of the model, or in this case the cost function. This iterated subtraction looks like this,

$$W = W - \eta \nabla C, \tag{2.7}$$

where W is the weights matrix in the MLP architecture, ∇C is the gradient of C(W), and η is the learning rate. This same iteration, for an specific weight in the MLP, could be written like this,

$$w_{ij}^{L} = w_{ij}^{L} - \eta \frac{dC}{dw_{ij}^{L}}, \tag{2.8}$$

where w_{ij}^{L} is the specific weight to be optimized, L is the number of layer in which the weight is encountered, i is the node for which the weight is connected in the next layer ($N(L+1)$) and j is the node from which the weight is connected ($N(L)$). Backpropagation is an algorithm that uses the gradient descent to minimize the cost in an ANN architecture in a way that is faster than previous learning approaches [28]. In Equation 2.6 it should be noticed that each h can be expressed in terms of its weights and inputs as seen on equations 2.1 and 2.2. In a MLP with one or more hidden layers the outputs h can be expressed in terms of the outputs and weights from the previous layers. Since MLP are nested activation functions. To calculate the derivative of a weight that is encountered at layer $(L-1)$ is necessary to use the chain rule,

$$\frac{dC}{dw_{ij}^{L-1}} = \frac{dC}{dh^{L}} \frac{dh^{L}}{dz^{L}} \frac{dz^{L}}{dw_{ij}^{L-1}}. \tag{2.9}$$

Backpropagation uses linear algebra tools such as elementwise vector multiplication to compute all the partial derivatives at once, instead of every individual partial derivative.

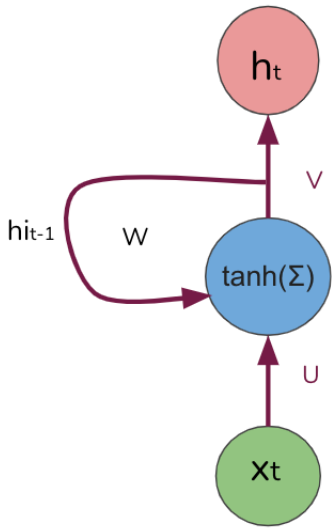


Figure 2.6: Graphic representation of a RNN node.

2.1.3 Recurrent Neural Networks

Another name that is given to the MLP are feed forward ANN (FFANN) because of the way the information propagates through the network without any feedback [20]. On the other hand recurrent neural networks (RNN) feedback the last output as an input in each node. This is useful to handle sequential data.

The RNN consists in a recurrent state called the hidden state (hi_t),

$$hi_t = tanh(Ux_t + Wh_{t-1}), \tag{2.10}$$

where U is the weight matrix for the input and W is the weight matrix for the previous hidden state h_{t-1}. The first hidden state is set as $hi_0 = 0$. Then the output of a RNN node is defined as

$$h_t = \sigma(Vh_{it}), \tag{2.11}$$

where V is the output weight matrix. A graphic representation of a RNN node is shown on Figure 2.6 where it can be seen how the information feedbacks on the RNN. To train RNNs, W needs to be backpropagated through all the time steps, this is called backpropagation through time (BPTT). To visualize the necessity of this, it can help to view the RNN in an unfolded way, as shown on Figure 2.7. Where is shown how W can be backpropagated

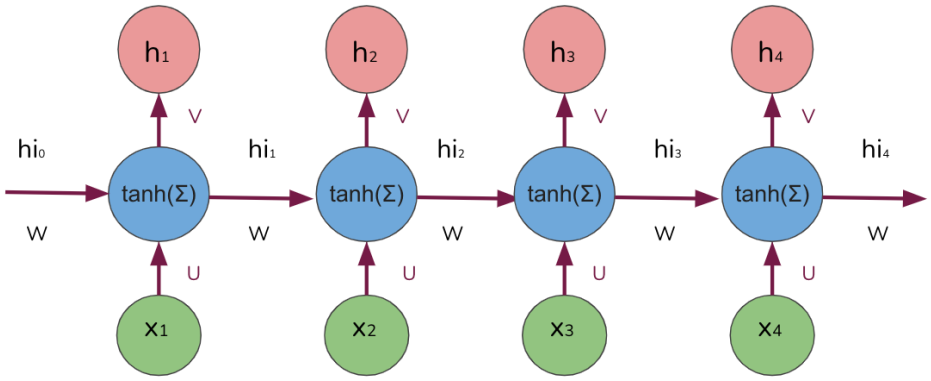


Figure 2.7: Graphic representation of an unfolded RNN.

until reaching time step $t = 1$. Nevertheless RNNs tend to have problems in its training process. Because of this feedback information, its gradient tend to increase exponentially or diminish until it stops learning from data [9]. This problem is called the exploding and vanishing gradient. In 1997, Sepp Hochreiter and Jürgen Schmidhuber [9] proposed a variety of RNN which they called Long-Short Term Memory (LSTM) to solve this problem.

Long-Short Term Memory Networks

LSTM difference with RNN relies in an additional recurrent state, called the cell state (C_t) [12] that works as a long term memory that keeps track of the important information [29]. Another difference is that the information is filtered by gates, being these the input(i_t), forget(f_t) and output(o_t) gates. The i_t decides what current information is going to pass through,

$$i_t = \sigma(x_t U^i + h_{t-1} W^i), \tag{2.12}$$

in which W^i and U^i are the respective weight matrix of the gate. The f_t decides the information that is no longer relevant to keep in the hidden and cell state,

$$f_t = \sigma(x_t U^f + h_{t-1} W^f). \tag{2.13}$$

The o_t decides the internal state information that needs to be passed,

$$o_t = \sigma(x_t U^o + h_{t-1} W^o). \tag{2.14}$$

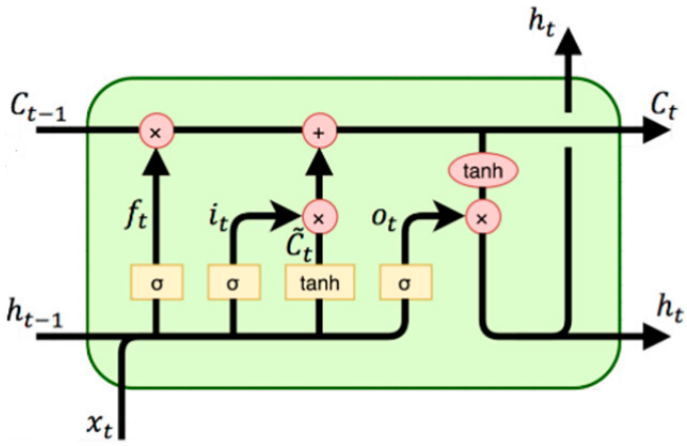


Figure 2.8: Graphic representation of an LSTM unit [20].

Using these gates, the recurrent states are calculated. C_t is obtained in two equations, at first is calculated a candidate for C_t ($\tilde{C}_t$) using:

$$\tilde{C}_t = tanh(x_t U^c + h_{t-1} W^c), \tag{2.15}$$

secondly the C_t is calculated using:

$$C_t = \sigma(f_t \odot C_{t-1} + i_t \odot \tilde{C}_t). \tag{2.16}$$

Finally to calculate h_t is used

$$h_t = tanh(C_t) \odot o_t. \tag{2.17}$$

In contrast with the RNN hidden state h_{it}, LSTM output and recurrent hidden state gets the same value of h_t. A graphic representation of how the information flows into a LSTM unit is shown on Figure 2.8. Where it can be noticed that i_t, f_t and o_t decide how much of the information passes to the cell state. At the same time the C_t determines the output and h_t of the unit for the next time step.

2.1.4 Convolutional Neural Network

In 1982 Fukushima and Miyake [30] proposed the predecessor of CNNs which was called 'Neocognitron' and it was based on the responsive properties of the visual cortical neurons. It was until 1990 were the first CNN was created by Yann LeCun [31].

Usually 2D CNN are used to extract local features from images [22]. However 1D CNN are mostly used and more efective to extract local features in a time series [32]. CNN differs with a MLP because its neurons are sparsely connected with weights between layers reducing the number of parameters that need to be trained. Also information passes through so called filters or convolutions, which can be understood as a sum of a weighted multiplication that slides trough the input features. This can be described with the next equation,

$$h_c = \sum_{i=1}^{N} x_i \otimes W, \tag{2.18}$$

where h_c is the output of the convolution, N is the number of input features, x_i are the input features and W is the weight matrix that, when trained, works as a feature filter. Then h_c passes through an activation function to become the output of the CNN unit (h),

$$h = f(h_c). \tag{2.19}$$

Another characteristic of CNNs is that between each convolution layer there is a pooling layer. This layer does not have any weight or trainable parameter. However is used to reduce the size of h, reducing the computing resources needed to train the model's parameters on forward layers [32].

Table 2.1 shows a summary of the main characteristics of the three kinds of ANN models presented in this section. Some common applications for each kind of ANN model are also shown there.

2.1.5 CNN-LSTM model

The model used in this work is based on the work made by Tovar et al [22] which is a hybrid ANN model that uses CNN layers for local feature learning and LSTM layers for temporal feature learning. This is built by just stacking the desired amount of layers of each kind. In the mentioned article it was proposed a model of five CNN layers, five LSTM layers and a MLP layer to give an output. This model showed to have better performance than a two CNN layers and two LSTM layers model , and better than five layer LSTM model for photovoltaic power forecasting. On Chapter 3 are tested more combinations of this hybrid architecture for global horizontal irradiance imputation.

ANN Model	Main Characteristics	Common Applications
FFANN	-Information flows forward. -Nodes are interconnected through consecutive layers.	- General regression problems.
RNN	-Information flows through recursive connections. - Each recursion is a hidden state.	- Time series prediction. -Natural language processing.
CNN	-Sparsely connected layers -Filters or Kernels slides through the input features.	- Image recognition and classification.

Table 2.1: Main characteristics and most common applications of the three kinds of ANN models.

2.2 Statistical models

Another kind of models that are commonly used for time series are the statistical ones, where simpler linear and non linear models are used. For this work a model called seasonal autoregressive integrated moving average (SARIMA) was used.

2.2.1 Time series statistical analysis

Time series in a statistical point of view can be understood as a sequential set of data points measured typically over successive times [33]. The analysis of time series is the procedure of fitting time series to a proper model. They can be decompose into four parts, being these ones the trend ($T(t)$), cyclical ($C(t)$), seasonal ($S(t)$) and the error or irregular ($E(t)$). Where $T(t)$ is the tendency of the time series to increase or decrease. $C(t)$ are when the time series presents rises and falls with a not fixed frequency [34]. $S(t)$ are the rises and falls that occur with a fixed frequency. $E(t)$ component is caused by unpredictable influences that are not regular or repeat in any particular pattern. These components can describe time series in two ways. An additive decomposition,

$$Y(t) = T(t) + C(t) + S(t) + E(t), \tag{2.20}$$

or a multiplicative decomposition,

$$Y(t) = T(t)C(t)S(t)E(t). \tag{2.21}$$

Additive decomposition is based on the assumption that the four components are independent of each other, while multiplicative decomposition is based on the assumption that the four components are not necessarily independent [33].

For statistical models it is assumed that time series follows a certain probability model. The probability structure of the time series is termed as a stochastic process [33]. An usual assumption is that time series variables are independent and identically distributed following the normal distribution, in other words that these are stationary. Stationarity for time series means that its properties do not depend of the time in which is observed [34]. This means that a time series with trend or seasonality is not stationary, while white noise is defined as stationary.

2.2.2 ARMA models

ARMA models consists in a sum of two models: autoregressive model (AR) and moving average model (MA). In an AR model, a linear combination of past values, also called lags, is used to predict the values of a variable. An AR(p) model or an AR model of order p is expressed as:

$$y_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i y_{t-i}, \tag{2.22}$$

where p is the number of lags, y_t is the variable to be predicted, c is the intercept, ϕ are the parameters that multiply the p lags and ϵ_t is the random error at t time step. In an AR(p) model is assumed that every y_t observation depends on the p lagged observations. This implies that y_t has an indirect dependency with lags larger than p and back to the very first value of the time series t_0. This recursive dependency on y_t lagged values makes the AR(p) model to be considered long memory models [35]

On the other hand moving average (MA) models uses past forecasted errors to predict the desired variable or also called error lags (ϵ_{t-q}). A MA(q) model or an MA model of order q is expressed as:

$$y_t = c + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}, \tag{2.23}$$

where q is the number of error lags and θ are the parameters that multiply the q error lags. Since in this model there is no a recursive dependency of y_t on error lags MA models are considered short memory models [35].

As mentioned before ARMA models is the sum of AR and MA models, so an ARMA(p,q) model can be expressed as:

$$y_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}. \tag{2.24}$$

It is important to acknowledge that ARMA models need to be fed with stationary data, this would make necessary to de-trend and de-season the time series.

2.2.3 Non-seasonal ARIMA models

Autoregressive integrated moving average (ARIMA) models introduces a differentiation order (d) to ARMA models, with the objective to eliminate or to reduce trend and seasonality on time series in order to make them stationary [34]. A differentiated series is the change between consecutive observations in the original series, a differentiated series with order $d = 1$ (y'_t) can be described as:

$$y'_t = y_t - y_{t-1}, \tag{2.25}$$

for a differentiated series with order $d = 2$ (y'') could be described as:

$$y''_t = y'_t - y'_{t-1}. \tag{2.26}$$

This way an ARIMA(p,d,q) model equation could be described as:

$$y_t^{(d)} = c + \epsilon_t + \sum_{i=1}^{p} \phi_p y_{t-p}^{(d)} + \sum_{i=1}^{q} \theta_q \epsilon_{t-q} \tag{2.27}$$

where $y_t^{(d)}$ is the d times differenced time series.

2.2.4 SARIMA models

Seasonal autoregressive integrated moving average (SARIMA) takes into account seasonal data. For this another set of seasonal orders $(P, D, Q)_m$ is added, where P, D and Q are the seasonal AR, I and MA order respectively and $_m$ is the number of time steps between seasons. A seasonal differentiated time series could be described as:

$$y'_t = y_t - y_{t-m}, \tag{2.28}$$

a SARIMA model is presented as $SARIMA(p, d, q)(P, D, Q)_m$. To stablish the optimal orders needed to model a time series is necessary to make an analysis of the autocorrelation and partial autocorrelation functions [33].

2.2.5 Autocorrelation and partial autocorrelation functions

Autocorrelation function (ACF) measures the relationship between the y_t and its k lagged value, y_{t-k} [34]. Mathematically an ACF coefficient (ρ_k) can be described as the rate between the covariance of y_t with y_{t-k} (γ_k),

$$\gamma_k = Cov(y_t, yt - k), \tag{2.29}$$

and the covariance with zero lags (γ_0), which is equivalent as the time series variance [33],

$$\rho_k = \frac{\gamma_k}{\gamma_0}. \tag{2.30}$$

On the other hand partial autocorrelation function (PACF) measures the relationship between the y_t and its k lagged value but removing the effects of lags $1, 2, 3, \cdots, k-1$ [34].

Since ACF and PACF bring us the relations of any y_t with its lagged value. Plotting these functions on several lags can bring us information of the optimal ARIMA$(p,q,d)(P,Q,D)_m$ orders to be used [36]. These plots are called corrrelograms. Certain behaviors on these correlograms could indicate specific signs to use different orders on the ARIMA model. For example if ACF and PACF coefficients are critical (larger than $\frac{1.96}{\sqrt{N}}$, where N is the number of observations) through most of the observed lags, is an indication to add a differentiation d order. After differentiating the time series, an ARIMA(p,d,0) sign would happen when critical values on ACF decays slowly, while there is a critical spike in the p lag on PACF. An ARIMA(0,d,q) sign would happen when critical values on PACF decays slowly, and ACF shows a critical spike in the q lag. These same signs can determine SARIMA seasonal orders $(P,D,Q)_m$, but in this case the spikes on the correlograms are presented on multiples of m lags. This is applied on Section 3.4 where a SARIMA$(p,d,q)(P,D,Q)_m$ orders are determined.

Chapter 3

Models training and comparison

This chapter presents the methodology to select the best global horizontal irradiance imputation model. Section 3.1 describes the data used to feed the models. Section 3.2 shows the metrics used to evaluate and compare the models. Section 3.3 presents the methodology to explore the hyperparameters for a Convolutional Long-Short Term Memory Neural Network (CNN-LSTM) model. Section 3.4 presents the methodology to find the optimal orders for a Seasonal Autoregressive Integrated Moving Average (SARIMA) model. Section 3.5 shows a comparison of the performance between CNN-LSTM and SARIMA.

3.1 Data description

The data used for this work is obtained from the Solarimetric and Meteorological Station in the Renewable Energies Institute (ESOLMET-IER) that is located in Temixco, Morelos in Mexico with a latitude 18°50'25.62"N, and longitude 99°14'10.49"W. The ESOLMET-IER measures the global horizontal irradiance (I_g) [$\frac{W}{m^2}$], direct normal irradiance (I_b) [$\frac{W}{m^2}$], diffuse solar irradiance (I_d) [$\frac{W}{m^2}$] and solar ultraviolet-A irradiance [$\frac{W}{m^2}$], as well as some common meteorological variables such as air dry bulb temperature (T_o) [^oC], relative humidity (rh) [%], wind speed [$\frac{m}{s}$] and barometric pressure [$mbar$]. However not all variables were used for this work as stated on Section 3.3.1.

The station reports every 10 minutes and has been working since 2010, this data can be seen on the ESOLMET-IER website with a previous registration. For this work years 2019, 2018 and a typical meteorological year

(TMY) has been used [37]. A TMY is a year built with measured data from the weather's most representative months of different years. These months are selected using a statistical analysis. Nevertheless, with every change of month, the TMY presents discontinuous data on meteorological variables, with exception on the irradiance variables. However the impact of these discontinuities are assumed as negligible for the purposes of this work.

3.1.1 Data split

Data of three different years is selected for training, validation and testing purposes respectively. The training year contains the data from which the model learns. Therefore for the CNN-LSTM this year was used to train the model. While for SARIMA, training year was used to determine the optimal model orders. Training data must be statistically representative, so the CNN-LSTM and the SARIMA model can generalize its predictions when new data is given. For this purpose the TMY of Temixco, presented in [37] was used. This TMY contains the typical solar irradiance months from 2010 to 2015.

The validation year is used for tuning the hyperparameter of the CNN-LSTM model. In contrast with training data, the CNN-LSTM model does not learn from it but is only for evaluating the performance of the model with data that the model has not been trained for. To accomplish this purpose the year 2019 was chosen, since it had fewer missing data spots than other years. Since several hyperparameters are being adjusted to decrease the error metrics on the validation year, is necessary to use another data set to confirm that the model is really generalizing on new data.

Testing data, as its name suggest is used to test the model once is already trained and tuned. The year 2018 was chosen for this purpose. However it should be cleared that 2018 has missing data, mainly in May. The data from May of year 2019 is used to impute this missing data, some other isolated missing days are imputed by pasting the previous days. Year 2018 is also used to create the weather file (epw) for the EnergyPlus simulations to be implemented, and to evaluate the impact of the imputations on Chapter 4.

3.2 Metrics

The next metrics are used to evaluate the performance of the models: The irradiance absolute error,

$$I_g^{err} = |I_g^p - I_g|, \tag{3.1}$$

where I_g^p is the predicted I_g. Which is used as the main metric to choose the best CNN-LSTM hyperparameter tuning model, seen on Section 3.3.2. As well as to compare the performance between CNN-LSTM and SARIMA models.

To view this error in daily energy terms the models are evaluated with the next metrics: The daily energy absolute error,

$$E_d^{err} = |E_d^p - E_d|, \tag{3.2}$$

expressed on $[\frac{Wh}{m^2}]$, where E_d^p is the daily predicted solar energy and E_d is the real daily solar energy. And the daily energy absolute percentage error

$$E_d^{\%err} = \frac{|E_d^p - E_d|}{E_d} \times 100. \tag{3.3}$$

3.3 CNN-LSTM training and tuning

The CNN-LSTM model was programmed using the high level application programming interface Keras and TensorFlow. The CNN-LSTM model is based on the model shown in [22]. However the model built in this work differs from the based model in its architecture, the data feeding process, as well as the output and some input variables.

3.3.1 Data feeding preparation

Before feeding the CNN-LSTM model it is necessary to prepare the data. The way to feed the model for its training is by setting our independent variable, or input data, as the present value. And setting our dependent variable, or target, in this case I_g, as the variable we want to predict but forward in time on any desired time horizon.

It was established to set a prediction time horizon of 1 day ahead. In the used weather data, there are 144 time steps per day, or 6 time steps per hour. The model would predict 6 time steps at once using the input data from 6 time steps in the day before. This process is iterated until the entire day is predicted. In comparison with training the model to predict 144 time steps at once, it was found that the proposed approach reduced significantly the computational time of training.

Input data on the CNN-LSTM model is exogenous, this means that it does not consist only on the past measurements of the target variable I_g, but also on the past values of other correlated variables, such as I_b, T_o and rh as well as temporal data that was added using solar altitude (α)

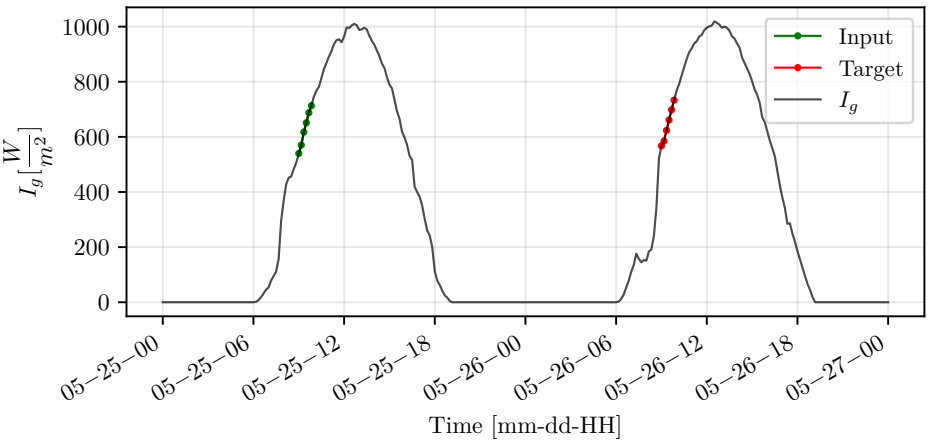


Figure 3.1: Graphic representation of I_g input and target in the CNN-LSTM model.

and azimuth (γ) angles, so the model could recognize the seasonal changes around the year. In other words, in order to predict I_g at any hour of tomorrow, the model should be fed with the weather and temporal data of the same hour from today as portrayed on Figure 3.1. Where the green dotted line represents the input of the I_g values, while the red dotted line is the target I_g values forward in time for 1 day. Even if is not represented on the figure, it should be remembered that input data contains also I_b, T_o, rh, α and γ values.

3.3.2 Hyperparameter tuning

Every Machine Learning (ML) based model, such as the CNN-LSTM model, has a large amount of possible hyperparameters that can be tuned. For this work a grid search was used. A grid search consists on trying every possible combination of hyperparameters to be tuned in order to find the best combination. As it can be seen on Table 3.1, three hyperparameters were chosen to be tuned: the number of CNN, LSTM and Feed Forward layers (CNN-LSTM-FFL), the learning rate (η), and the batch size (BS). By the end of this grid search, twenty seven different hyperparameters combinations were evaluated, then the best of this options can be selected.

The metrics, shown on Section 3.2, were evaluated on all the validation year by calculating the annual mean value of them. However the annual mean value of I_g^{err} was used to determine the best and the worst model

CNN-LSTM-FFL # of layers	η [-]	BS [-]
3-3-1	1e-7	6
		12
		24
	1e-5	6
		12
		24
	1e-4	6
		12
		24
2-2-1	1e-7	6
		12
		24
	1e-5	6
		12
		24
	1e-4	6
		12
		24
1-1-1	1e-7	6
		12
		24
	1e-5	6
		12
		24
	1e-4	6
		12
		24

Table 3.1: Hyperparameter grid search with the CNN, LSTM and feed forward layers on the ANN architecture, CNN-LSTM-FFL, learning rate, η, and batch size, BS, combinations.

CNN-LSTM-FFL	η [-]	BS [-]	E_d^{err} $[\frac{Wh}{m^2}]$	$E_d^{\%err}$ [%]	I_g^{err} $[\frac{W}{m^2}]$
2-2-1	$1e^{-4}$	12	539.5	10.2	80.1
1-1-1	$1e^{-7}$	24	720.3	12.2	98.4

Table 3.2: Table of the best and worst performance model, with their training hyper parameters; number of layers, learning rate, η, and batch size, BS, and their respective metrics values annual mean daily energy absolute error, E_d^{err}, annual mean daily energy absolute percentage error, $E_d^{\%err}$, and the irradiance mean absolute error, I_g^{err} .

on the grid search. Also it was determined that the night values, when α has a negative value, would be discarded since this would underestimate the metric and giving the impression that the error of the model was smaller than it actually was. The best grid search model, with the smallest I_g^{err}, had a value of $80.1\frac{W}{m^2}$, an E_d^{err} of 539.5 $\frac{Wh}{m^2}$ and a $E_d^{\%err}$ of 10.2%. The worst model, with the largest I_g^{err}, had a value of $98.4\frac{W}{m^2}$, an E_d^{err} of $720.3\frac{Wh}{m^2}$ and a $E_d^{\%err}$ of 12.2%. The reason for $E_d^{\%err}$ value being so close in both models, with a difference of only 2%, can be explained because the best grid search model tend to overestimate the E_d^p with a greater magnitude than the worst grid search model when E_d has low values. On Table 3.2 are presented the hyperparameters of these models and its metrics values for each of them. On Table 3.3 is presented the detailed architecture of the best CNN-LSTM model with their correspondent number of filters on CNN layers, hidden states on LSTM and nodes on Dense layers, as well as the activation function that was used on each layer.

Models performance

On Figure 3.2 is presented the comparison of I_g^p, I_g and instantaneous I_g^{err} for (a) the worst performance day and (b) the best performance day of the best model in the validation year. It can be observed that the worst performance day seems to be a cloudy and probably a rainy day due to irregular and small I_g that results into the model overestimating its I_g^p. On this day the model reaches a day mean $I_g^{err} = 175.1\frac{W}{m^2}$, an $E_d^{err} = 4194.9\frac{Wh}{m^2}$ and a $E_d^{\%err} = 200.2\%$. Meanwhile the best performance day seems to be a clear sky day due to the large I_g values and results into the model having a very good performance. On this day the model reaches a day mean $I_g^{err} = 7.2\frac{W}{m^2}$, an $E_d^{err} = 84.7\frac{Wh}{m^2}$ and a $E_d^{\%err} = 1.2\%$.

Layer	Filters/Hidden states/Nodes	Activation function
Input	6	None
1D-CNN	64	Relu
Pooling	-	None
1D-CNN	64	Relu
Pooling	-	None
LSTM	128	Tanh
LSTM	64	Tanh
Dense	6	Sigmoid

Table 3.3: Best grid search architecture, where each row represents a layer, with its correspondent number of filters, hidden states or nodes and its activation function.

On Figure 3.3 is presented the comparison of I_g^p and I_g for (a) the worst performance day and (b) the best performance day of the worst model in the validation year. It can be observed that the worst performance day is the same day as on Figure 3.2(a) and it seems that the model also overestimates the value of the I_g. On this day the model reaches a day mean $I_g^{err} = 145.9\frac{W}{m^2}$, an $E_d^{err} = 3500.9\frac{Wh}{m^2}$ and a $E_d^{\%err} = 166.9\%$. Meanwhile the best performance day although it seems also as a cloudy day the relatively good performance can be explained due to the similar conditions that the prior day had, and the lack of sudden changes of I_g. On this day the model reaches a day mean $I_g^{err} = 16.5\frac{W}{m^2}$, an $E_d^{err} = 229.4\frac{Wh}{m^2}$ and a $E_d^{\%err} = 5.4\%$.

To give an insight of the performance of the selected models across the different seasons of the year, as well as to acknowledge the conditions in which these models performed the best and the worst a I_g^{err} daily mean calendar heatmap is shown on Figure 3.4 for (a) the best and (b) the worst CNN-LSTM model on the search grid. And on 3.4(c) is shown the calendar heatmap of the E_d. Each square of these calendar heatmaps, divided by white lines, represents a day in the year. Each section divided by black lines represents a month. Every column of squares represents a week starting by Monday in the upper square and ending on Sunday in the lower square of the column. By comparing these three figures is clear to see how both models seems to decrease its performance between the months of June and October. It can also be observed that the worst performance days coincide with days with a low E_d that are preceded with higher E_d days. As it can be seen in the beginnings of June and ending of September.

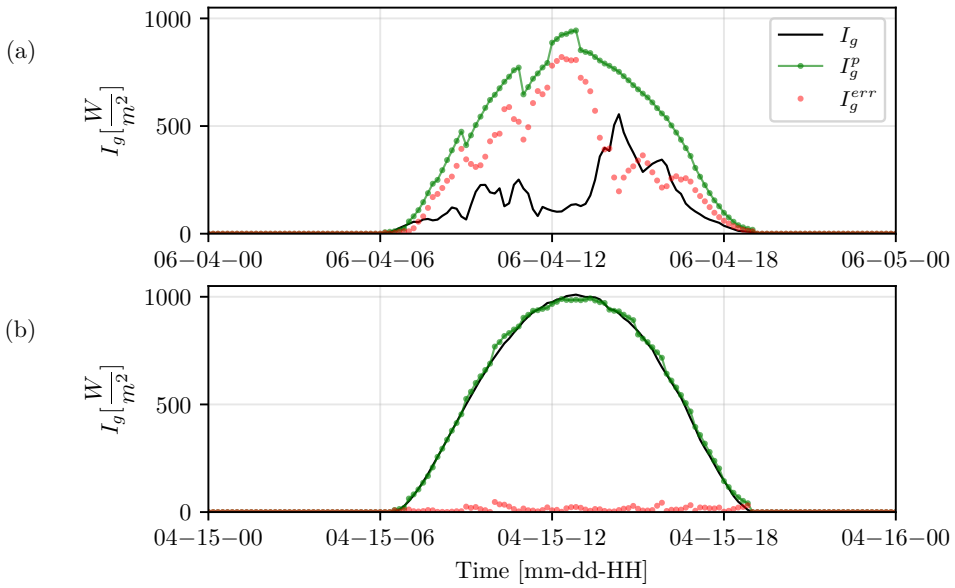


Figure 3.2: Best model comparison of I_g^p, I_g and instantaneous I_g^{err} for (a) the worst performance day and (b) the best performance day, in the validation year.

3.4 SARIMA model order determination

SARIMA models, in contrast with CNN-LSTM model, is usually used with less frequent measurements (annual or monthly time steps) and does not need to be trained with the data of an entire year. Since this would take a lot of computational time and does not guarantee the model to be more accurate. For this reason, the so called training year was only used to make the analysis to determine the orders of the model. However SARIMA model needs to be trained before each predicted day with the previous five days. SARIMA model orders can be determined via interpretation of Auto Correlation Function (ACF), and Partial Auto Correlation Function (PACF) correlograms [36]. The first step is to identify the differentiation order (d) and the seasonal differentiation order (D). As mentioned on Section 2.2.3 differentiation consists on subtracting the lagged value of the time series with a lag of one, for a non seasonal differentiation, and a lag of m for a seasonal differentiation, where m is the number of time steps of the season. By plotting the PACF, on Figure 3.5 (a), and ACF on Figure 3.5(b) of the I_g with the training year, it can be observed that there are consistent spikes

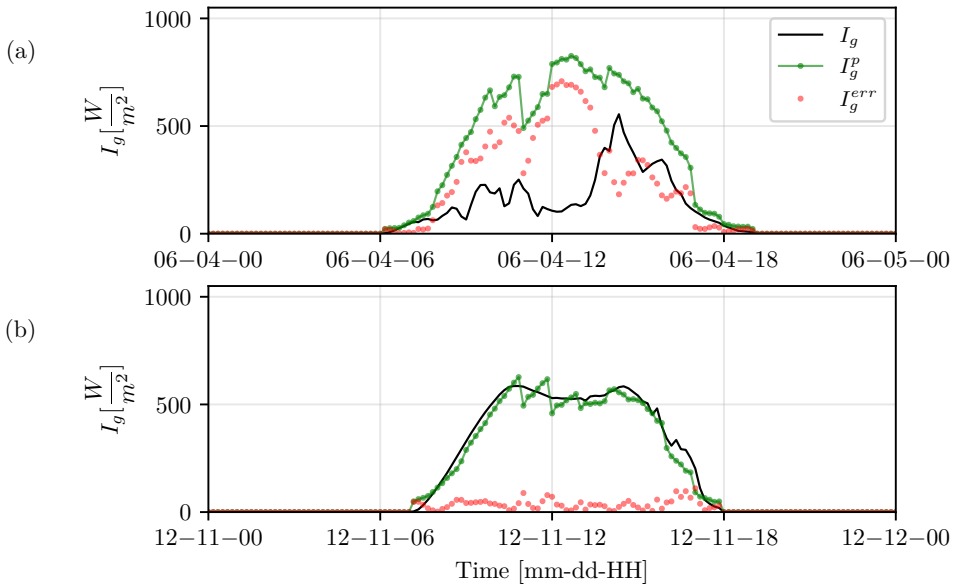


Figure 3.3: Worst model comparison of I_g^p, I_g and instantaneous I_g^{err} for (a) the worst performance day and (b) the best performance day, in the validation year.

on the ACF plot. This indicates that it should be used at least one order of non seasonal differentiation (d).

After non seasonal differentiating, PACF and ACF are plotted again, shown on Figure 3.6(a) and Figure 3.6(b) respectively. It can be observed on ACF that there are critical spikes every 144 lags. Which indicates it needs at least one seasonal differentiation order (D).

The next step is to identify the number of SAR (P) and SMA (Q) orders. By looking at the seasonal and non seasonal differentiated correlograms: PACF on Figure 3.7(a) and ACF on Figure 3.7(b). On PACF graph can be seen there are several spikes in every 144 lag that decays slowly. On the other hand, on ACF graph can be seen a single significant spike on lag 144. These characteristics indicates to add one SMA (Q) order. Since is not recommended to have both SMA(Q) and SAR(P) orders at the same time to avoid overfitting. The orders of the model to be used were SARIMA$(0,1,0)(0,1,1)_{144}$. This model would then be used to predict the test year and compare its performance against the CNN-LSTM model on Section 3.5. The model will be trained using the previous five days of each predicted day.

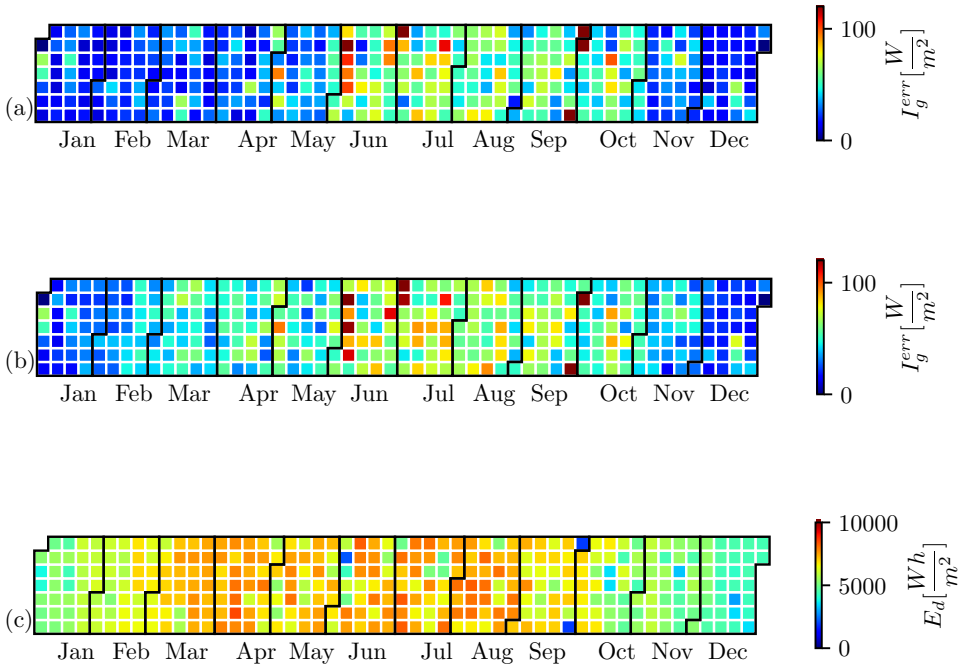


Figure 3.4: Irradiance daily mean absolute error I_g^{err} calendar heatmap for (a) The best CNN-LSTM model. (b) The worst CNN-LSTM model. Also it is shown (c) the daily solar global radiation calendar heatmap.

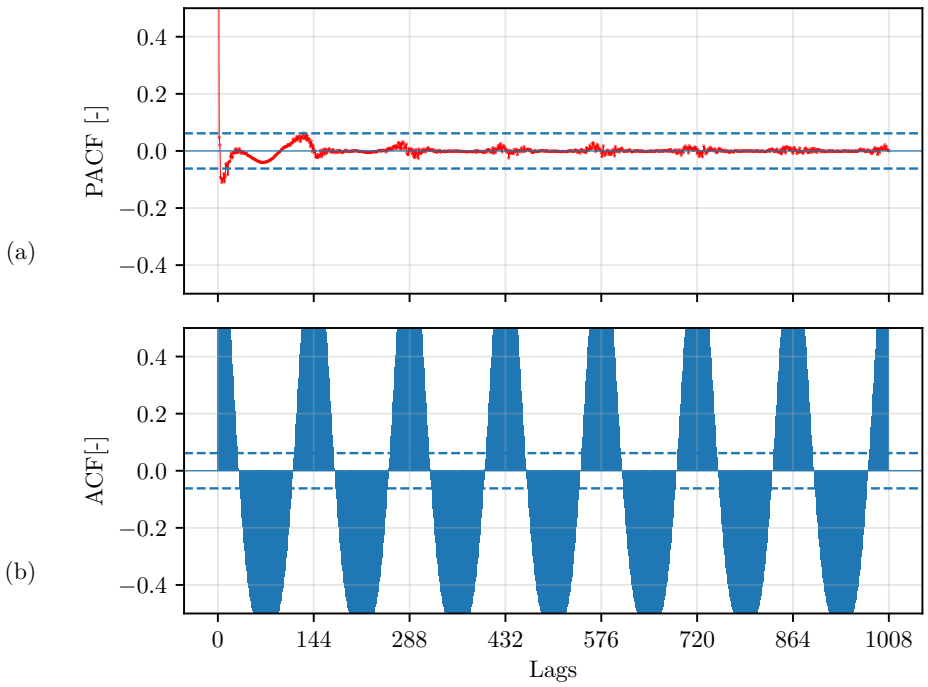


Figure 3.5: (a) PACF graph and (b) ACF graph of training year I_g.

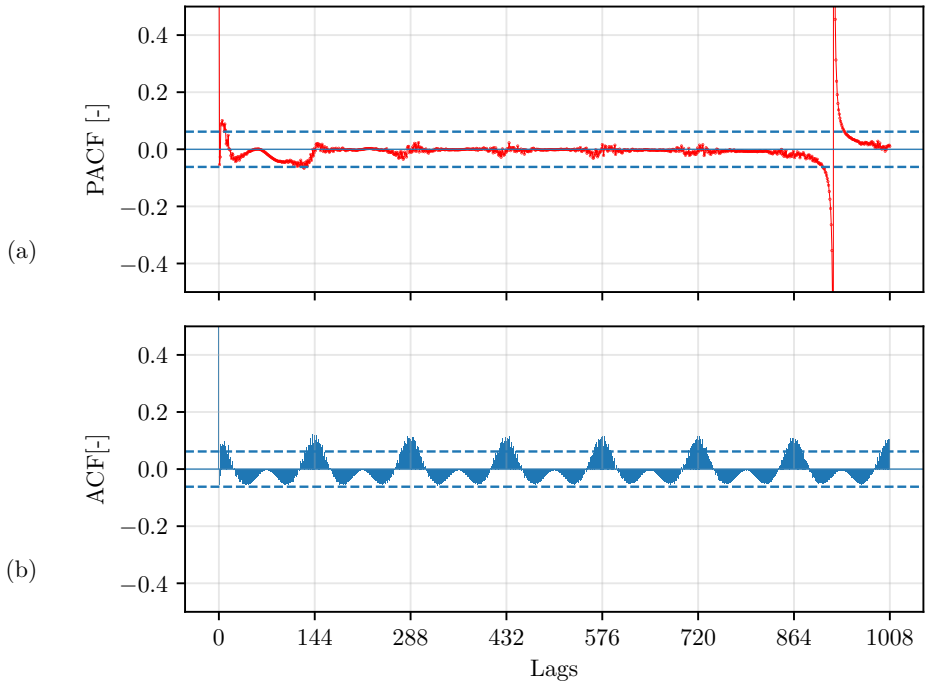


Figure 3.6: (a)PACF graph and (b)ACF graph of training year I_g, after a non seasonal differentiating.

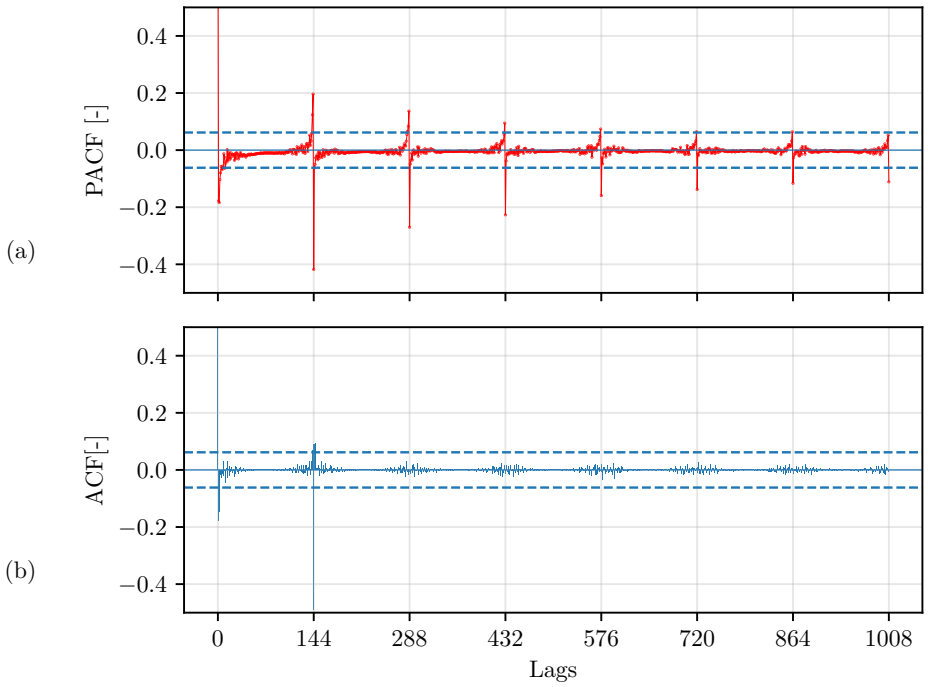


Figure 3.7: (a)PACF graph and (b)ACF graph of training year I_g, after non seasonal and seasonal differentiation.

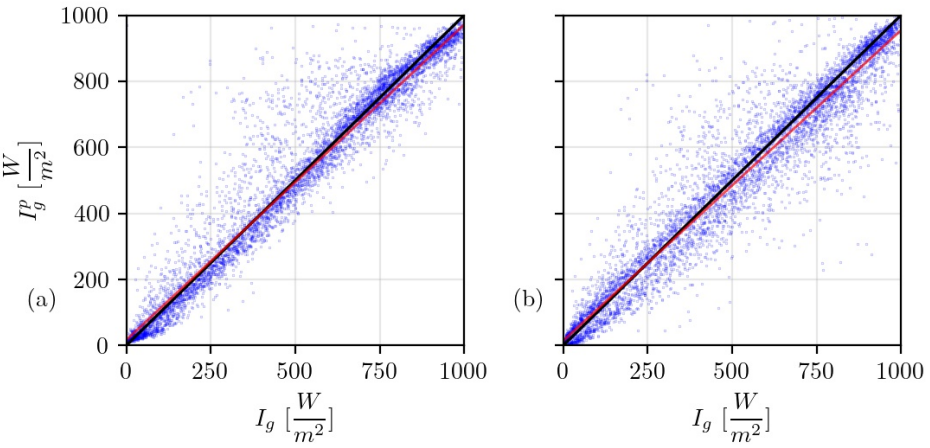


Figure 3.8: Linear adjustment of the I_g and I_g^p obtained from (a) the CNN-LSTM model, and (b) SARIMA model. Where the red line is the line that resulted from the linear regression, while the black line represents the linear regression for a perfect match.

3.5 Models comparison

After getting SARIMA orders, and training CNN-LSTM model the test year was taken to compare the performance of both of these models. The comparison was carried out by the year mean value of E_d^{err}, $E_d^{\%err}$ and I_g^{err} to determine which model has a better performance to impute. Also the values of the slope (m) and the intercept (b) of a linear adjustment that corresponds with the equation $I_g^p = mI_g + b$, and the correlation coefficient (R^2) of the same linear adjustment were used as additional metrics for this comparison.

On Figure 3.8 is presented the linear fit of the comparison between I_g and I_g^p for (a) the CNN-LSTM model and (b) the SARIMA model. Where it can be observed that the CNN-LSTM model gets closer to the perfect match, in comparison with SARIMA model in which can be observed that I_g^p reaches lower values than the CNN-LSTM model, indicating that there is a slight tendency of SARIMA model to underestimate its predictions. However, CNN-LSTM model seems to overestimate its predictions. Nevertheless, quantitatively R^2 values for both models are very close to each other. SARIMA presents a $R^2 = 0.93$ while CNN-LSTM model presents a $R^2 = 0.94$ which may indicate that both models has a similar performance.

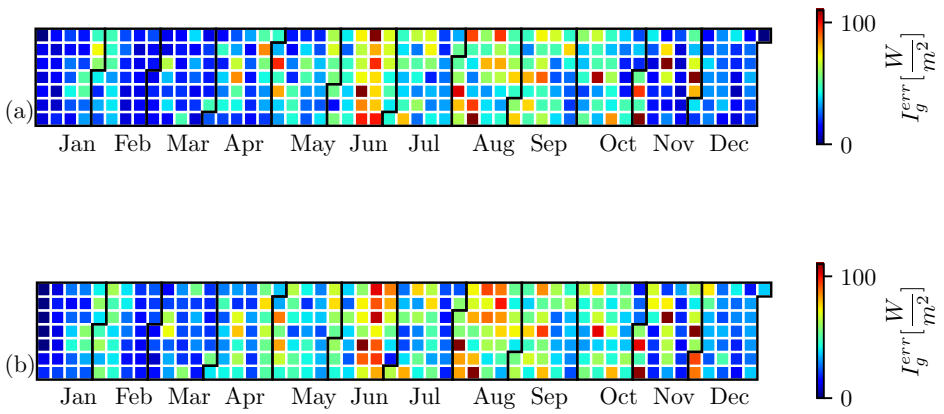


Figure 3.9: Calendar heatmap of the daily I_g^{err} of the test year for (a) the CNN-LSTM model, and (b) the SARIMA model.

The calendar heatmap of the daily mean I_g^{err} is shown on Figure 3.9 for (a) the CNN-LSTM model and (b) the SARIMA model to get an overview of the performance of each model. On these calendar heatmaps can be noticed that CNN-LSTM model presents more critical high I_g^{err} days. However the error propagates for fewer days than in SARIMA model. Which results in the CNN-LSTM having lower I_g^{err} days than SARIMA model. The larger propagation of error of SARIMA model in comparison with CNN-LSTM model is caused by the need of SARIMA model to be trained with the last five days before prediction. While CNN-LSTM model only needs a prior day to make the prediction. This affirmation is more noticeable on June and November.

On Table 3.4 are shown the metrics. Where is confirmed that the performance of both models is similar. Nevertheless the CNN-LSTM model outgrades the SARIMA model in all metrics, with an exception on b. For this reason the CNN-LSTM model is used on Chapter 4 to evaluate its impact on a building simulation made on EnergyPlus.

Model	E_d^{err} $[\frac{Wh}{m^2}]$	$E_d^{\%err}$ [%]	I_g^{err} $[\frac{W}{m^2}]$	m [-]	b $[\frac{W}{m^2}]$	R^2 [-]
CNN-LSTM	530.0	11.2	76.3	0.95	16.5	0.94
SARIMA	606.6	12.3	86.4	0.94	14.5	0.93

Table 3.4: Comparison of CNN-LSTM and SARIMA model performance with the established metrics annual mean daily energy absolute error, E_d^{err}, annual mean daily energy absolute percentage error, $E_d^{\%err}$, and the annual mean irradiance absolute error, I_g^{err}, as well as the linear regression slope, m,the linear regression intercept, b, and the correlation of determination, R^2, metrics.

Chapter 4

Impact of imputations on the thermal behavior of a building

On the last chapter was selected the best model to impute the global horizontal irradiance based on several metrics, the selected model was the Convolutional Long-Short Term Memory (CNN-LSTM), which proved to have a better performance than the Seasonal Autoregressive Integrated Moving Average model (SARIMA). In this chapter is presented the impact of the imputed global horizontal irradiance on the zone mean air temperature of the thermal zones of a building simulation on EnergyPlus.

On Section 4.1 are described the characteristics of the simulated building using EnergyPlus. On Section 4.2 is presented the criteria used to choose the critical days when global horizontal irradiance is imputed. On Section 4.3 are described the metrics used to evaluate the impact of imputed global horizontal irradiance in comparison with measured global horizontal irradiance on the zone mean air dry bulb temperature of the thermal zones of a building simulated with EnergyPlus. This last section ends with an analysis of the behavior of the metrics.

4.1 Building EnergyPlus simulation

The model of the building is built on Sketchup, and was developed by the energy in buildings investigation group in the Renewable Energies Institute, UNAM. The simulated building is located at Temixco Morelos, in a latitude 18.84^o N and a longitude 99.26^o W less than one kilometer away from the

(a)

(b)

Figure 4.1: NELIER realistic render view of (a) the South façade and (b) the North façade.

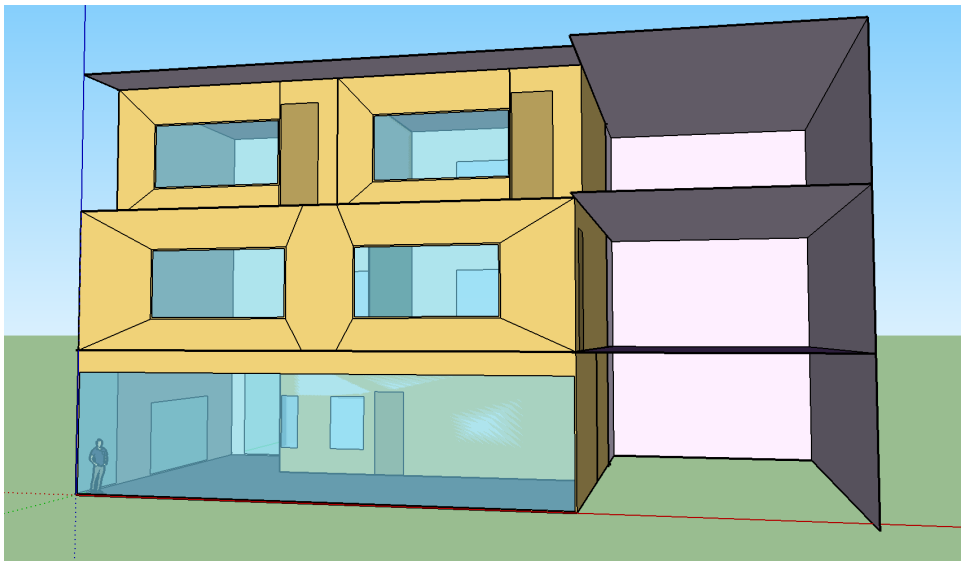
Solarimetric and Meteorological Station from the Renewable Energies Institute (ESOLMET-IER). The study case is on construction and incorporates bioclimatic strategies. In this thesis it will be called NELIER (Nuevo Edificio de la LIER). Figures 4.1(a) and 4.1(b) show the South and the North façade respectively. NELIER consists on three floors and a subterranean parking site, it has two staircases one on the East side of the building and another one at the center which divides the building into two sections. The South façade of the West and the East section are shown on Figure 4.2(a) and (b) respectively. West section has five thermal zones conformed by laboratories and a dining room. East section has twenty five thermal zones, mostly conformed by classrooms and offices.

For the purposes of this work it was considered to use the East section of NELIER for the evaluation of its thermal zones, since it is in this section where people will be most of the time. Four thermal zones were discarded to evaluate since these were on its majority not used for large amounts of time as is the case of a hallway on the second floor. Or are totally inaccessible for people, as is the case of three thermal zones that will have tubes passing between the ground floor and the first floor. The mentioned staircases at the center and on the East side of the building are represented with shading surfaces. The subterranean parking site will be represented on the simulation by setting an outdoor boundary condition at the floor, with no sun exposure.

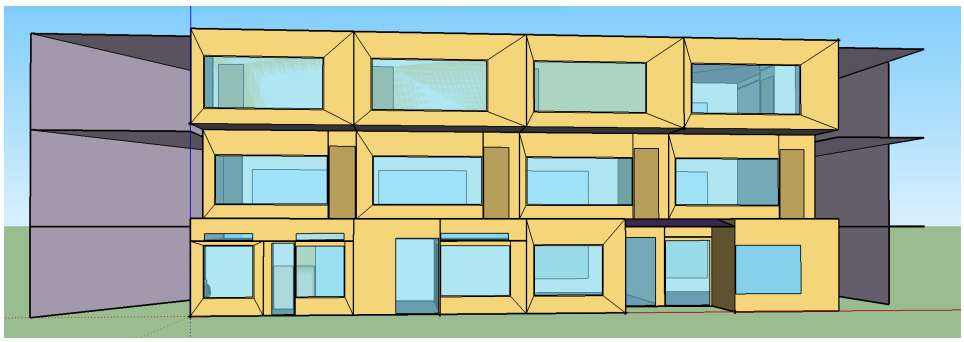
The simulation takes into account an air infiltration of 0.5 air changes per hour, with no air conditioning and it has thermal loads from people depending on the expected occupation schedules. Solar protections are not implemented in the SketchUp model. Two simulations were done. The first simulation was done using the weather data of the test year (2018), described in Section 3.1.1, and is named the base simulation (BS). The second simulation was done by selecting specific days in this same year and imputing the horizontal global irradiance $(I_g)[\frac{W}{m^2}]$ using the CNN-LSTM model. This one is named the imputed simulation (IS). The specific selected days, as well as the followed methodology is on Section 4.2.

4.2 Methodology for imputed days selection.

CNN-LSTM can make a day ahead prediction of I_g, by using weather data of the day before. This makes possible to build a year made only with predicted I_g $(I_g^p)[\frac{Wh}{m^2}]$ by using each day of the measured data year. This enables a comparison between I_g and I_g^p, that are used to select the critical imputed days and its impact on the building simulation. For this procedure



(a)



(b)

Figure 4.2: NELIER building Sketchup model South façade of (a) West section and (b) East section.

the test year is used.

1. The CNN-LSTM model is used to make a daily prediction of all the year, this year was called the imputed year, since all the I_g were replaced by I_g^p values, by removing one day at a time.

2. The values of the test year and the imputed year are evaluated using the daily mean of the irradiance absolute error I_g^{err} using the Equation 3.1.

3. The daily I_g^{err} values of all the year are then separated into four parts, each one of 3 months long (January-March, April-June, July-September, October-December). The days with the largest and smallest daily mean I_g^{err} for each of the year's parts were chosen to be deleted from the test year and replaced with I_g^p values from the imputed year.

This procedure resulted into the test year having eight imputed days, four with the worst, largest I_g^{err} and four with the best, smallest I_g^{err} days. On Table 4.1 are shown the days that had the largest or worst I_g^{err} for the four parts of the year. As well as the metrics used on Chapter 3, like the the daily energy absolute error (E_d^{err}), the daily energy absolute percentage error ($E_d^{\%err}$), determined by Equations 3.2 and 3.3 respectively, as well as the linear regression slope (m), bias (b) and correlation of determination (R^2). The largest I_g^{err} reached a value of 165.4 $\frac{W}{m^2}$ on 2018-11-14. As it was anticipated on the last chapter the days with largest I_g^{err} were also the ones with the smallest daily solar energy (E_d) on the season, which could indicate that the model performance tend to be worse on sudden cloudy and probably rainy days.

On the other hand the days with the smallest I_g^{err} tend to happen when there is a similar E_d in both the input and the target day. On Table 4.2 are presented the days with the smallest or best I_g^{err} for the imputation with the CNN-LSTM model for the four parts of the year. As well as the other metrics previously mentioned E_d^{err}, $E_d^{\%err}$, m, b and R^2 in which the smallest I_g^{err} was 5 $\frac{W}{m^2}$ on 2018-01-12.

4.3 Impact evaluation of the imputation

To evaluate the impact that imputation has on a thermal simulation, the zone mean air temperature (T_i) $[^oC]$ is compared between the BS and the IS. To quantitatively evaluate this impact the next metrics are used.

Date	E_d^{err} $[\frac{Wh}{m^2}]$	$E_d^{\%err}$ [%]	I_g^{err} $[\frac{W}{m^2}]$	m [-]	b $[\frac{W}{m^2}]$	R^2 [-]
2018-01-30	1388.9	36.2	70.7	1.2	27.6	0.84
2018-06-15	3560.6	110.4	149.3	1.6	69.9	0.76
2018-08-12	3423.8	102.7	142.6	1.4	89.0	0.76
2018-11-14	3917.8	405.4	165.4	4.4	26.3	0.75

Table 4.1: Selected days with the worst metrics; daily energy absolute error, E_d^{err}, daily energy absolute percentage error, $E_d^{\%err}$, irradiance absolute error , I_g^{err}, linear regression slope, m, linear regression intercept, b, and linear regression coefficient of determination, R^2.

Date	E_d^{err} $[\frac{Wh}{m^2}]$	$E_d^{\%err}$ [%]	I_g^{err} $[\frac{W}{m^2}]$	m [-]	b $[\frac{W}{m^2}]$	R^2 [-]
2018-01-12	14.2	0.2	5.0	1.0	-1.4	0.99
2018-04-17	128.2	1.8	13.5	1.0	5.9	0.99
2018-07-19	291.1	3.7	17.1	0.9	2.1	0.99
2018-12-21	23.0	0.4	5.57	1.0	0.4	0.99

Table 4.2: Selected days with the best metrics; daily energy absolute error, E_d^{err}, daily energy absolute percentage error, $E_d^{\%err}$, irradiance absolute error , I_g^{err}, linear regression slope, m, linear regression intercept , b, and linear regression coefficient of determination, R^2.

Temperature absolute error,

$$T_i^{err} = |T_i^p - T_i|, \tag{4.1}$$

where T_i^p is the temperature reported by IS, while T_i is the temperature reported by BS. Where the instantaneous T_i^{err} values are used to be evaluated by the next metrics.

Vanishing time,

$$V_t = t(threshold) - t(imputation), \tag{4.2}$$

where $t(imputation)$ is the time when imputation is started, at sunrise, and $t(threshold)$ is the time when T_i^{err} is less or equal a determined threshold, which was set to 0.1^oC. V_t is expressed on days hh:mm and measures the time that the effect of the imputation lasts on T_i until reaching a negligible impact.

Maximum temperature absolute error,

$$T_{err}^{max} = max(T_i^{err}), \tag{4.3}$$

where T_{err}^{max} is the maximum value that T_i^{err} reaches after the imputation. Only one value of T_{err}^{max} is obtained for each imputation. T_{err}^{max} is expressed on ^oC and measures the magnitude of the maximum T_i^{err} that is reached in each imputation. T_{err}^{max} is useful to know the expected maximum error that an imputation can have when using the CNN-LSTM model on a simulation.

Maximum error time,

$$max_t = t(T_{err}^{max}) - t(imputation), \tag{4.4}$$

where $t(T_{err}^{max})$ is the time when T_{err}^{max} occurs. max_t is expressed on days hh:mm and measures the time that T_i^{err} takes to reach T_{err}^{max}.

To summarize, the three metrics V_t, T_{err}^{max} and max_t are helpful to know the nature of the impact of the imputation on T_i. V_t measures the length of time that the imputation has on T_i. T_{err}^{max} measures the maximum magnitude that T_i^{err} reaches. max_t measures the point in time where T_{err}^{max} occurs.

For every thermal zone of NELIER, eight T_{err}^{max} are calculated using the imputed days presented on Table 4.1 and 4.2. The mean value for these eight days of T_{err}^{max} are calculated for all thermal zones. The thermal zones with the largest and smallest mean T_{err}^{max} of each floor were chosen for a more detailed evaluation.

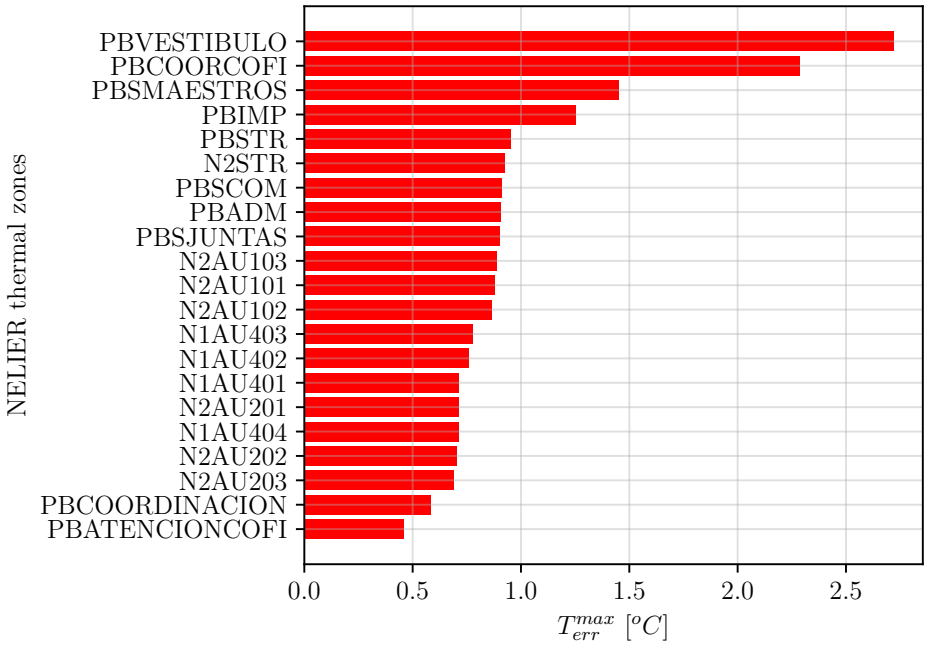


Figure 4.3: Thermal zones sorted by its eight imputations averaged T_{err}^{max} value in descending order.

On Figure 4.3 are shown the thermal zones sorted by the mean value of its T_{err}^{max} for the eight imputed days. The prefix in the name of thermal zones indicates the floor in which this is encountered. PB for the ground floor, N1 for the first floor and N2 for the second floor. As it is stated with more detail on Section 4.3.3 the magnitude of T_{err}^{max} is determined by the thermal zone oscilation of T_i.

4.3.1 Largest T_{err}^{max} thermal zones.

The thermal zones with the largest mean T_{err}^{max} per floor were N2STR, N1AU403 and PBVESTIBULO showed on Figures 4.4(a), 4.4(b) and 4.4(c) respectively, viewed from a South East perspective on NELIER SketchUp model.

Since I_g values are corrected to zero on the night time, the beginning of the imputing data is not started until sunrise just at the hour that is reported on the Date imputed column of the next tables.

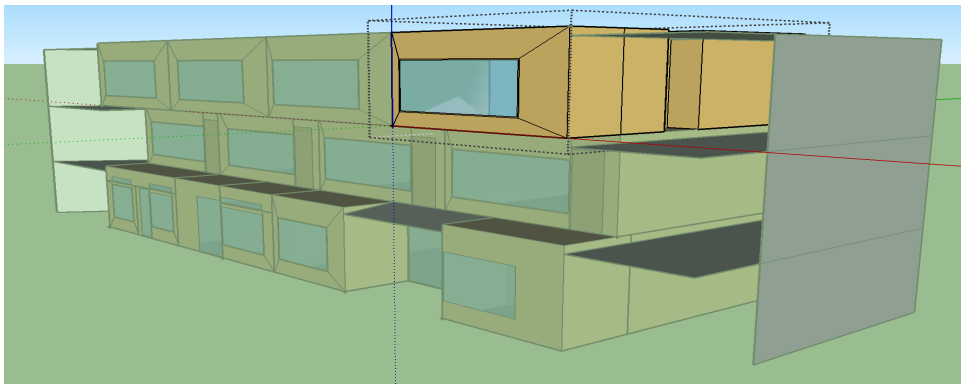
On Table 4.3 are presented the metrics V_t, T_{err}^{max} and max_t for the worst imputations on N2STR. The largest T_{err}^{max} was 2.8 ^oC on 2018-06-15. The maximum V_t was 6 days 11:40h on a different day on 2018-11-14. The largest max_t was 10:40 h after the imputation on 2018-08-12. The smallest T_{err}^{max} was 0.8 ^oC on 2018-01-30. The minimum V_t was 4 days 01:00 on the same day on 2018-01-30. The smallest max_t was 08:30h after the imputation on 2018-11-14.

On Table 4.4 are presented the metrics V_t, T_{err}^{max} and max_t for the best imputations on N2STR. Where three out of four days had a V_t of 0 days 00:00 since T_{err}^{max} never was larger than the threshold of 0.1 ^oC on these days with the exception of day 2018-07-19 where T_{err}^{max} was 0.1 ^oC.

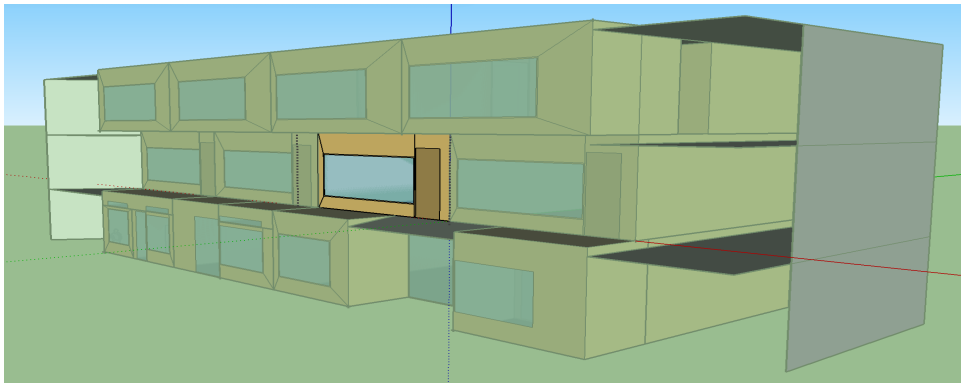
On Table 4.5 are presented the metrics V_t, T_{err}^{max} and max_t for the worst imputations on N1AU403. The largest T_{err}^{max} was 2.0 ^oC on 2018-11-14. The maximum V_t was 6 days 11:40h on a the same day on 2018-11-14. The largest max_t was 10:30 h after the imputation on 2018-08-12. The smallest T_{err}^{max} was 0.8 ^oC on 2018-01-30. The minimum V_t was 4 days 05:10 on the same day on 2018-01-30. The smallest max_t was 09:10h after the imputation on 2018-11-14.

On Table 4.6 are presented the metrics V_t, T_{err}^{max} and max_t for the best imputations on N1AU403. Where two out of four days had a V_t of 0 days 00:00 since T_{err}^{max} never was larger than the threshold of 0.1 ^oC, the largest T_{err}^{max} was 0.2 ^oC on 2018-07-19.

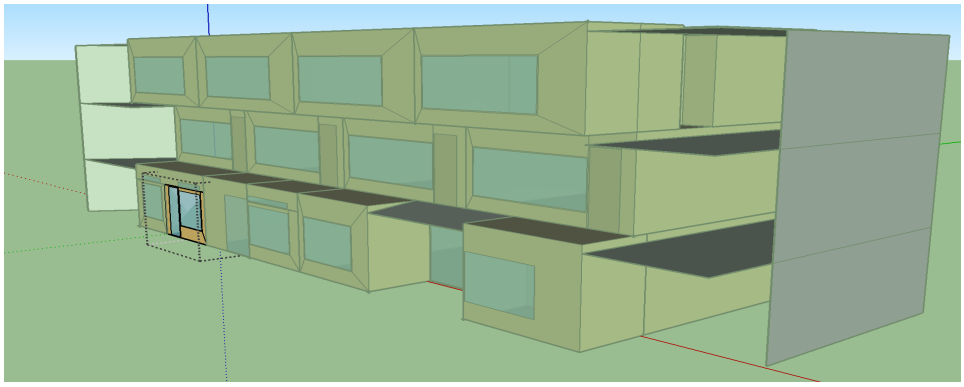
On Table 4.7 are presented the metrics V_t, T_{err}^{max} and max_t for the worst imputations on PBVESTIBULO. The largest T_{err}^{max} was 8.4 ^oC on 2018-11-



(a)



(b)



(c)

Figure 4.4: Thermal zones with the largest T_{err}^{max} of the year for each floor, view from a South East perspective (a) N2STR, (b) N1AU403 and (c) PB-VESTIBULO.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-30 07:20	4 days 01:00	0.8	0 days 09:40
2018-06-15 06:10	6 days 03:00	2.8	0 days 09:10
2018-08-12 06:30	5 days 15:20	1.3	0 days 10:40
2018-11-14 06:50	6 days 11:40	2.1	0 days 08:30

Table 4.3: Worst imputation days for thermal zone N2STR and its metrics; vanishing time, V_t, maximum error, T_{err}^{max} and maximum error time, max_t.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-12 07:20	0 days 00:00	0.0	0 days 01:00
2018-04-17 06:30	0 days 00:00	0.0	0 days 10:30
2018-07-19 06:20	0 days 18:00	0.1	0 days 08:00
2018-12-21 07:10	0 days 00:00	0.0	0 days 11:10

Table 4.4: Best imputation days for thermal zone N2STR metrics; vanishing time, V_t, maximum error, T_{err}^{max} and maximum error time, max_t.

14, the maximum V_t was 3 days 23:50h on a the same day on 2018-11-14. This V_t was shorter than the thermal zones of upper floor on the same day. The largest max_t was 09:50 h after the imputation on 2018-06-15. The smallest T_{err}^{max} was 3.7 ^oC on 2018-01-30. The minimum V_t was 2 days 18:30 on the same day on 2018-01-30. The smallest max_t was 04:40h after the imputation on 2018-11-14.

On Table 4.8 is presented the metrics V_t, T_{err}^{max} and max_t for the best imputations on PBVESTIBULO. Where none of the four days had a V_t of 0 days 00:00. The largest T_{err}^{max} was of 0.4 ^oC on 2018-04-17 and 2018-07-19.

As it can be noticed V_t is larger on N1AU403 and N2STR in compar-

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-30 07:20	4 days 05:10	0.8	0 days 09:50
2018-06-15 06:10	6 days 08:40	1.6	0 days 10:30
2018-08-12 06:30	5 days 22:50	1.4	0 days 10:30
2018-11-14 06:50	6 days 14:30	2.0	0 days 09:10

Table 4.5: Worst imputation days for thermal zone N1AU403 metrics; vanishing time, V_t, maximum error, T_{err}^{max} and maximum error time, max_t.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-12 07:20	0 days 00:00	0.0	0 days 01:00
2018-04-17 06:30	0 days 00:00	0.0	0 days 10:40
2018-07-19 06:20	0 days 15:50	0.2	0 days 07:50
2018-12-21 07:10	0 days 18:20	0.1	0 days 11:10

Table 4.6: Best imputation days for thermal zone N1AU403 metrics; vanishing time, V_t, maximum error, T_{err}^{max}, and maximum error time, max_t, on the 1st floor.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-30 07:20	2 days 18:30	3.7	0 days 09:00
2018-06-15 06:10	3 days 17:10	4.5	0 days 09:50
2018-08-12 06:30	3 days 12:20	3.8	0 days 04:40
2018-11-14 06:50	3 days 23:50	8.4	0 days 06:50

Table 4.7: Worst imputation days for thermal zone PBVESTIBULO metrics; vanishing time, V_t, maximum error, T_{err}^{max} and maximum error time, max_t, on the ground floor.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-12 07:20	0 days 19:10	0.2	0 days 11:00
2018-04-17 06:30	0 days 20:20	0.4	0 days 10:30
2018-07-19 06:20	0 days 18:00	0.4	0 days 07:20
2018-12-21 07:10	0 days 19:00	0.3	0 days 01:00

Table 4.8: Best imputation days for thermal zone PBVESTIBULO metrics; vanishing time, V_t, maximum error, T_{err}^{max}, and maximum error time, max_t, on the ground floor.

ison with PBVESTIBULO. On the other side T_{err}^{max} tends to be larger on PBVESTIBULO, in comparison with N1AU403 and N2STR.

On Figure 4.5(a),(b) and (c) is shown the comparison of T_i of the IS and BS for the evaluated thermal zones N2STR, N1AU403 and PBVESTIBULO respectively. On Figure 4.5(d) is presented a visualization of the T_i^{err} evolution at each time step during the V_t of each thermal zone where it can be seen the behavior of the metrics previously mentioned. These graphs belongs to the same day, 2018-11-14, since this is the day that presented the worst I_g^{err}.

4.3.2 Smallest T_{err}^{max} thermal zones

The thermal zones with the smallest T_{err}^{max} per floor were N2AU203, N1AU404 and PBATENCIONCOFI showed on Figures 4.6(a), 4.6(b) viewed from a North West perspective and 4.6(c) viewed from South West perspective. On Table 4.9 are presented the metrics V_t, T_{err}^{max} and max_t for the worst imputations on N2AU203. The largest T_{err}^{max} was 1.5 ^oC on 2018-11-14. The maximum V_t was 7 days 14:20h on a the same day on 2018-11-14. The largest max_t was 10:30 h after the imputation on 2018-08-12. The smallest T_{err}^{max} was 1.0 ^oC on 2018-01-30. The minimum V_t was 4 days 23:10 on the same day on 2018-01-30. The smallest max_t was 09:10h after the imputation on 2018-11-14.

On Table 4.10 are presented the metrics V_t, T_{err}^{max} and max_t for the best imputations on N2AU203. Where three out of four days had a V_t of 0 days 00:00 since T_{err}^{max} never was larger than the threshold of 0.1 ^oC, the largest T_{err}^{max} was of 0.1 ^oC on 2018-07-19.

On Table 4.11 are presented the metrics V_t, T_{err}^{max} and max_t for the worst imputations on N1AU404. The largest T_{err}^{max} was 1.8 ^oC on 2018-11-14. The maximum V_t was 6 days 04:20h on a the same day on 2018-11-14. The largest max_t was 10:40 h on 2018-08-12. The smallest T_{err}^{max} was 0.7 ^oC on 2018-01-30. The minimum V_t was 3 days 22:30 on the same day on 2018-01-30. The smallest max_t was 09:10h after the imputation on 2018-11-14.

On Table 4.12 are presented the metrics V_t, T_{err}^{max} and max_t for the best imputations on N1AU404. Where three out of four days had a V_t of 0 days 00:00 since T_{err}^{max} never was larger than the threshold of 0.1 ^oC, the largest T_{err}^{max} was of 0.1 ^oC on 2018-07-19.

On Table 4.13 are presented the metrics V_t, T_{err}^{max} and max_t for the worst imputations on PBATENCIONCOFI. The largest T_{err}^{max} was 1.2 ^oC on 2018-11-14. The maximum V_t was 4 days 04:30h on a the same day on 2018-11-14. The largest max_t was 11:30 h on 2018-08-12. The smallest T_{err}^{max}

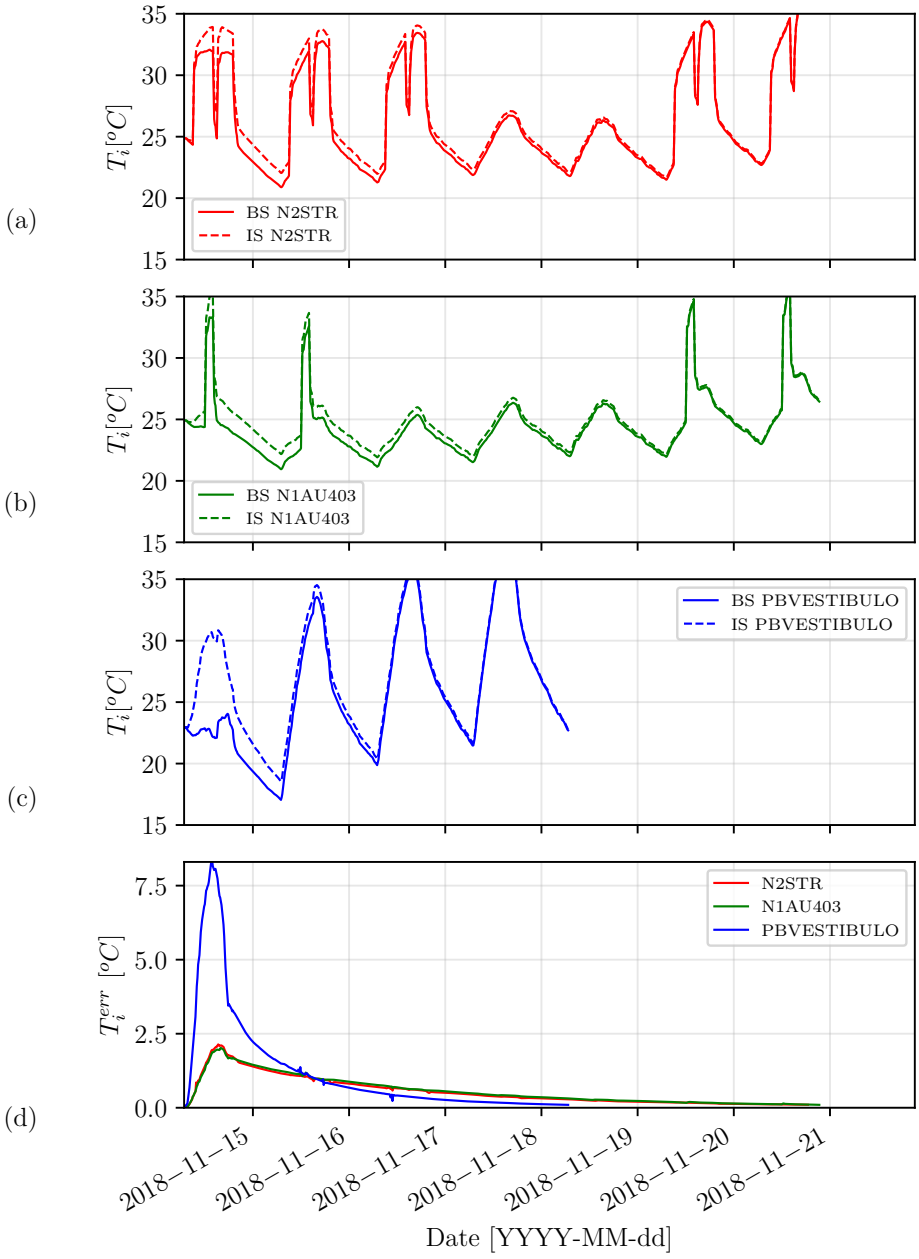


Figure 4.5: Comparison of the IS and BS T_i of (a) N2STR, (b) N1AU403 and (c) PBVESTIBULO. Also (d) T_i^{err} for the three thermal zones.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-30 07:20	4 days 23:10	1.0	2 days 03:00
2018-06-15 06:10	7 days 05:20	1.5	0 days 10:30
2018-08-12 06:30	6 days 22:10	1.2	0 days 10:30
2018-11-14 06:50	7 days 14:20	1.5	0 days 09:10

Table 4.9: Worst imputation days for thermal zone N2AU203 metrics; vanishing time, V_t, maximum error, T_{err}^{max}, and maximum error time, max_t.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-12 07:20	0 days 00:00	0.0	0 days 01:00
2018-04-17 06:30	0 days 00:00	0.0	0 days 10:30
2018-07-19 06:20	0 days 18:00	0.1	0 days 07:50
2018-12-21 07:10	0 days 00:00	0.0	0 days 11:10

Table 4.10: Best imputation days for thermal zone N2AU203 metrics; vanishing time, V_t, maximum error, T_{err}^{max}, and maximum error time, max_t.

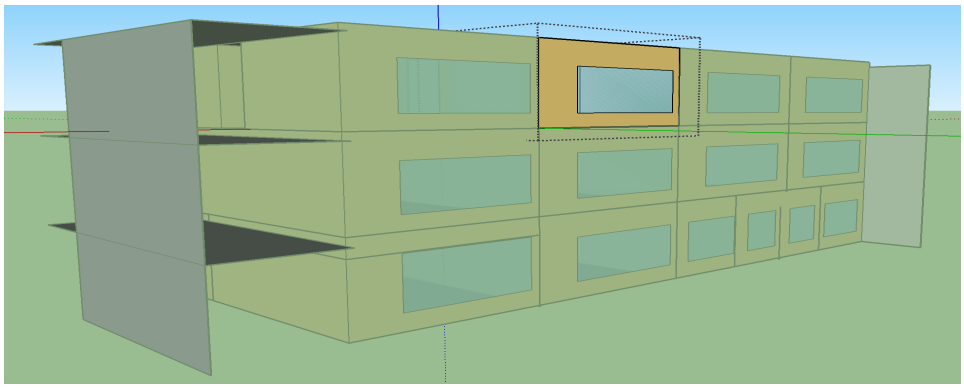
was 0.5 ^oC on 2018-01-30. The minimum V_t was 2 days 13:40 on the same day on 2018-01-30. The smallest max_t was 09:10h after the imputation on 2018-06-15.

On Table 4.14 are presented the metrics V_t, T_{err}^{max} and max_t for the best imputations on PBATENCIONCOFI. Where four out of four days had a V_t of 0 days 00:00 since T_{err}^{max} never was larger than the threshold of 0.1 ^oC.

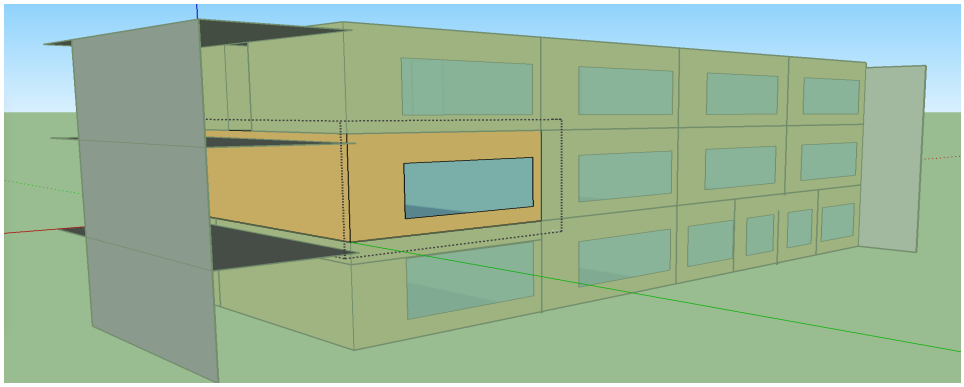
On Figures 4.7 (a), (b) and (c) is shown a comparison between the BS and IS T_i for N2AU203, N1AU404 and PBATENCIONCOFI respectively. As well on Figure 4.7(d) is presented a visualization of T_i^{err} evolution at each time step during the V_t of each thermal zone. These graphs belongs

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-30 07:20	3 days 22:30	0.7	0 days 09:50
2018-06-15 06:10	5 days 22:40	1.5	0 days 10:20
2018-08-12 06:30	5 days 08:40	1.3	0 days 10:40
2018-11-14 06:50	6 days 04:20	1.8	0 days 09:10

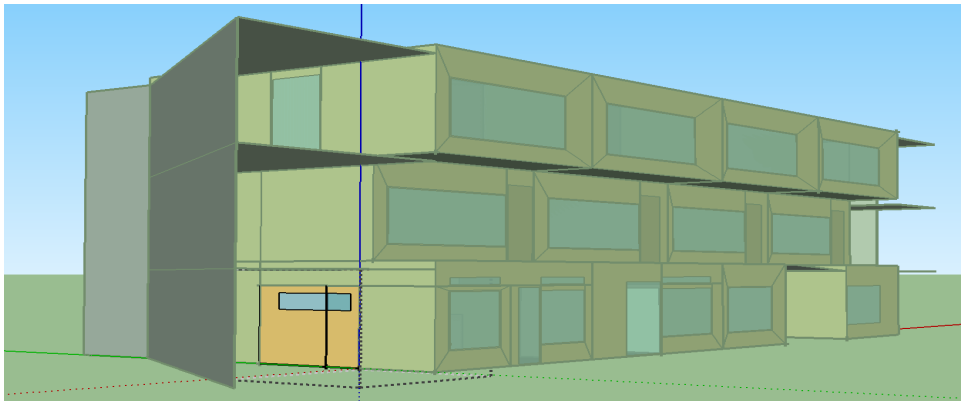
Table 4.11: Worst imputation days for thermal zone N1AU404 metrics; vanishing time, V_t, maximum error, T_{err}^{max}, and maximum error time, max_t.



(a)



(b)



(c)

Figure 4.6: Thermal zones with the smallest T_{err}^{max} of the year for each floor (a) N2AU203 and (b) N1AU404 viewed from a North East perspective. And (c)PBATENCIONCOFI viewed from a South West perspective.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-12 07:20	0 days 00:00	0.0	0 days 01:00
2018-04-17 06:30	0 days 00:00	0.0	0 days 10:30
2018-07-19 06:20	0 days 18:10	0.1	0 days 07:50
2018-12-21 07:10	0 days 00:00	0.0	0 days 11:10

Table 4.12: Best imputation days for thermal zone N1AU404 metrics; vanishing time, V_t, maximum error, T_{err}^{max}, and maximum error time, max_t.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-30 07:20	2 days 13:40	0.5	0 days 09:50
2018-06-15 06:10	3 days 17:50	1.0	0 days 09:10
2018-08-12 06:30	3 days 12:50	0.8	0 days 11:30
2018-11-14 06:50	4 days 02:30	1.2	0 days 09:30

Table 4.13: Worst imputation days for thermal zone PBATENCIONCOFI metrics; vanishing time, V_t, maximum error, T_{err}^{max} and maximum error time, max_t.

Date imputed	V_t [days hh:mm]	T_{err}^{max} [^oC]	max_t [days hh:mm]
2018-01-12 07:20	0 days 00:00	0.0	0 days 09:50
2018-04-17 06:30	0 days 00:00	0.0	0 days 10:30
2018-07-19 06:20	0 days 00:00	0.0	0 days 08:00
2018-12-21 07:10	0 days 00:00	0.0	0 days 06:50

Table 4.14: Best imputation days for thermal zone PBATENCIONCOFI metrics; vanishing time, V_t, maximum error, T_{err}^{max} and maximum error time, max_t.

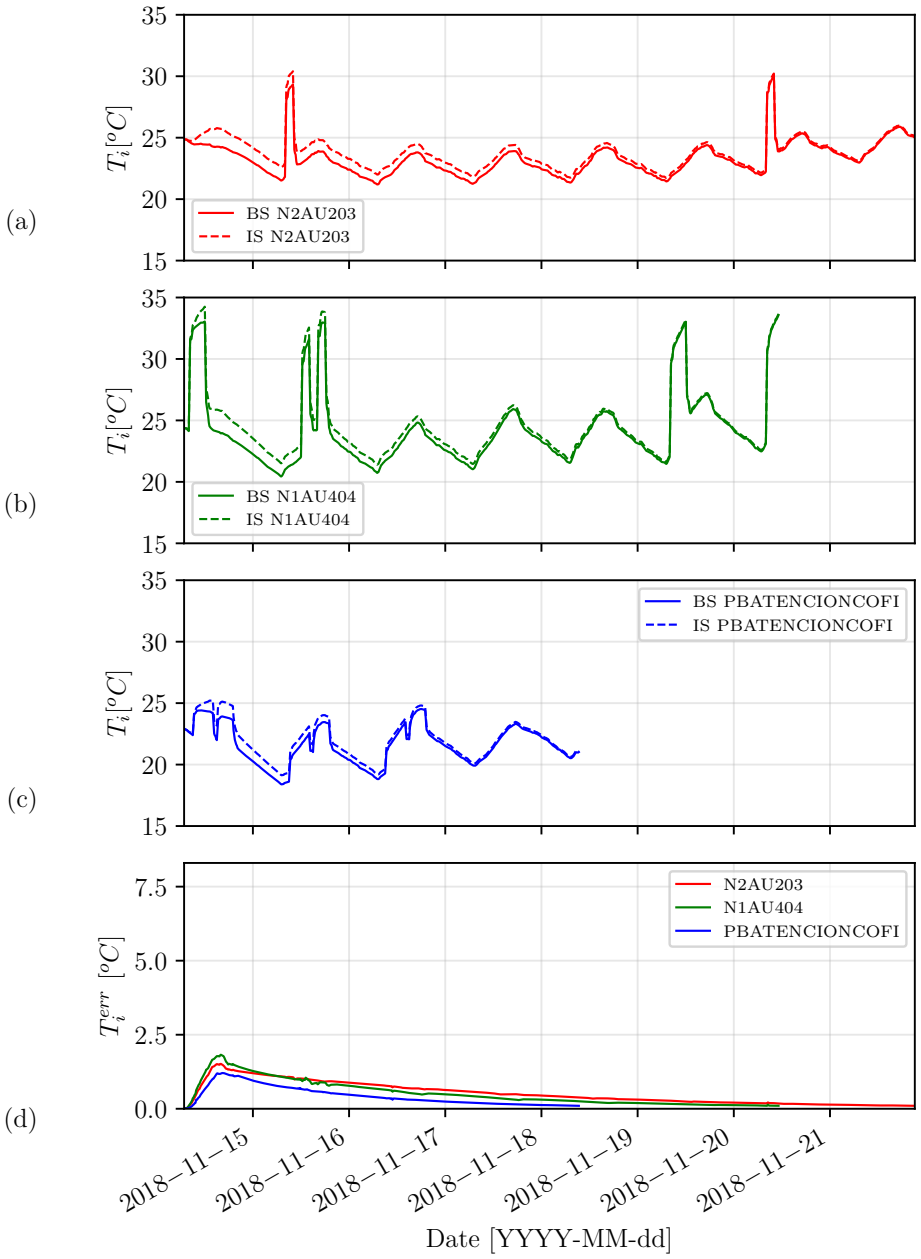


Figure 4.7: Comparison of imputed and base simulation T_i of (a) N2AU203, (b) N1AU404 and (c) PBATENCIONCOFI. Also (d) T_i^{err} for the three thermal zones.

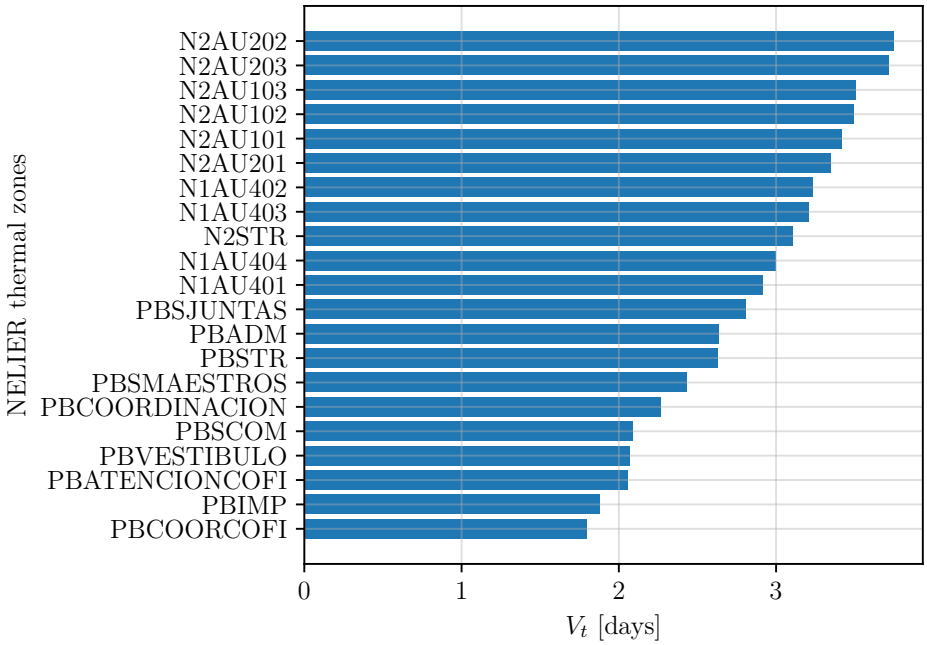


Figure 4.8: Thermal zones sorted by the eight imputations averaged V_t value in descending order.

to the same day, 2018-11-14, since this is the day that presented the worst I_g^{err}.

4.3.3 Metrics analysis

From the previous figures and metrics tables presented on the last section surged some observations and tendencies. The first one is the tendency of the ground floor thermal zones to show a smaller V_t than the ones of higher floors. On Figure 4.8 thermal zones are sorted by its averaged V_t and can be confirmed that, in most cases, the ground floor thermal zones has the smallest V_t, followed by the ones on the first floor and being the ones of the second floor with the largest V_t. This can be explained due to the I_g independent energy transfer that the ground floor has in comparison with higher floors. Which causes the imputation to have a less lasting disturbance than the higher floors, after the imputed day has ended.

The other observation comes from max_t values. The majority of the evaluated thermal zones had frequent values between 0 days 07:00 h and 0

days 11:30 h. There were two outliers of 01:00 h, that happened when T_{err}^{max} had a very small value on N2AU203 and N1AU403. And a large outlier of 2 days 03:00 h on N2AU203. However the frequent value of max_t indicates that T_{err}^{max} tends to occur around 16:00 and 18:00 hours of the day, which would be the same time when the maximum temperature of the thermal zone occurs if there was not any thermal load on it. This happens because of the tendency of the CNN-LSTM model to overestimate the I_g on cloudy days.

Lastly it was noticed that there is a slight tendency from T_{err}^{max} to get larger, when the yearly mean T_i oscillation (T_{osc}) is larger too. T_{osc} is given by,

$$T_{osc} =< T_{d\_max} - T_{d\_min} >, \tag{4.5}$$

where T_{d_max} is the maximum daily temperature and T_{d_min} is the minimum daily temperature reached at the interior of the thermal zone of the BS. Some factors, such as the volume and thermal isolation of thermal zones affects the magnitude of T_{err}^{max}. Nevertheless T_{err}^{max} is mainly determined by the incidence of I_g inside the thermal zone, and by how much this incident I_g affects T_i, affecting T_{osc} as well. This can explain the reason for PB thermal zones being in the highest and lowest positions on Figure 4.3. For example PBATENCIONCOFI, the lowest T_{err}^{max} thermal zone shown on Figure 4.4(c), is covered almost completely from I_g. While PBVESTIBULO, the highest T_{err}^{max} thermal zone shown on Figure 4.6(c), is a small thermal zone, with a window on the South façade which makes it being exposed to I_g. However the linear correlation between T_{osc} and T_{err}^{max},

$$T_{err}^{max} = 0.19 T_{osc} - 0.16^{o}C, \tag{4.6}$$

for the evaluated thermal zones on the given days begins to disperse after T_{osc} exceeds $7^{o}C$. On Figure 4.9 is shown the linear correlation model where each dot represents a thermal zone, the resulting coefficient of determination R^2 was of 0.42.

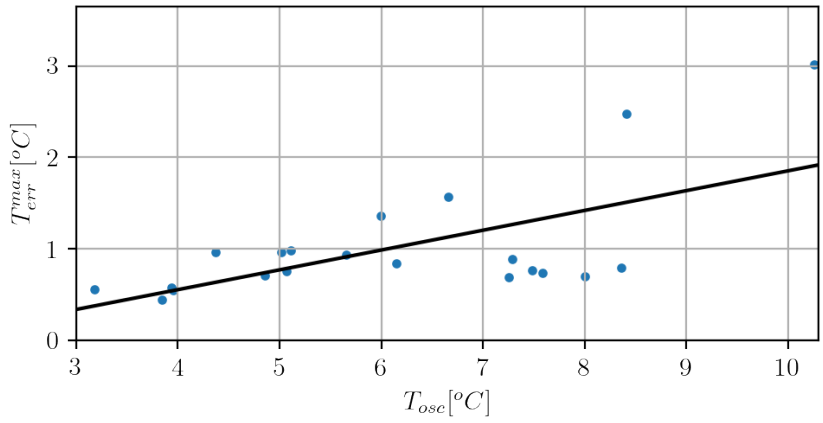


Figure 4.9: Linear correlation between T_{osc} and T_{err}^{max}.

Chapter 5

Conclusions

This chapter presents the conclusions obtained from the two main parts in which the work is divided. Section 5.1 presents a summary and conclusions from the comparison in the performance of imputation of global horizontal irradiance (I_g) between two models. Being these models a Convolutional Long Short Term Memory Artificial Neural Network (CNN-LSTM) and a Seasonal Autoregressive Integrated Moving Average (SARIMA). Section 5.2 presents the conclusions from the evaluation of the impact that CNN-LSTM model imputation has on the thermal behavior of a building simulated on EnergyPlus.

5.1 Models comparison

The CNN-LSTM model is set to predict the day ahead I_g using the previous day of metereological data as an input. Input meterological data includes variables such as direct normal irradiance (I_b), air drybulb temperature (T_o), relative humidity (rh) and I_g, as well as some temporal data variables where the solar altitude (α) and azimuth (γ) angles are used. SARIMA model is also set to predict the day ahead I_g. However SARIMA needs the previous five days of I_g values as an input.

The comparison is done using the metrics on Section 3.2. The annual mean absolute error in global horizontal irradiance (I_g^{err}) is the absolute difference between measured I_g and predicted I_g (I_g^p). I_g^{err} is used as the main metric to choose the best model. The annual mean of the daily irradiance energy error (E_d^{err}) is the absolute difference between the measured daily solar energy (E_d) and the predicted daily solar energy (E_d^p). E_d^{err} is used to express the prediction error in daily energy terms. The annual mean per-

centage error of the daily solar energy ($E_d^{\%err}$) is also used to evaluate the models error. Also the slope (m), bias (b) and correlation of determination (R^2) of a linear regression with I_g as the independent variable and I_g as the dependent variable are also used to evaluate the error.

The evaluated metrics of the CNN-LSTM model are $I_g^{err} = 76.3\frac{W}{m^2}$, $E_d^{err} = 530.0\frac{Wh}{m^2}$, $E_d^{\%err} = 11.2\%$, $m = 0.95$, $b = 16.5\frac{W}{m^2}$ and $R^2 = 0.94$. While the evaluated metrics of SARIMA are $I_g^{err} = 86.4\frac{W}{m^2}$, $E_d^{err} = 606.0\frac{Wh}{m^2}$, $E_d^{\%err} = 12.3\%$, $m = 0.94$, $b = 14.5\frac{W}{m^2}$ and $R^2 = 0.93$. It is noticed with these metrics that CNN-LSTM outgrades SARIMA model in all metrics except on b. However quantitatively the values are close to each other. This indicates that performance of both models is similar. Taking these characteristics into account some conclusions are made.

Even if CNN-LSTM outgrades SARIMA in almost all metrics, the quantitative advantage of CNN-LSTM is barely better than SARIMA. Considering the time needed to implement both models, SARIMA can be more advantageous than CNN-LSTM if the imputation model needs to be build from the ground up. However, when imputations need to be done repeatedly, CNN-LSTM can be more advantageous than SARIMA, because of the time it takes each model to achieve one day imputation. While CNN-LSTM can make an imputation almost instantaneously, SARIMA can take around five minutes for each imputation. Table 5.1 summarizes the advantages and disadvantages identified in this work when both models are compared with each other.

Future research works can focus on measuring the performance that models SARIMA and CNN-LSTM have when trying to predict for more than 1 day ahead time horizon. CNN-LSTM might have a significantly better performance for being a non-linear model. While SARIMA performance might decay sharply for being a linear model.

5.2 Impact of imputations on thermal behavior of a building

Solely taking into account the evaluated metrics on the last section, CNN-LSTM model is selected to evaluate the impact of its imputations on the thermal behavior of a building simulated on EnergyPlus. To evaluate this impact, I_g values from eight days of a year are purposely deleted so then they can be replaced with CNN-LSTM imputed I_g values. A simulation is done with the testing year (BS), and other simulation is done with the testing year but with these eight imputed days (IS). Then the zone mean air

Model	Advantages	Disadvantages
CNN-LSTM	- Needs only the previous day data to make the imputation. - Has a slightly better performance. - Each imputation is done almost instantaneously.	- More time-intensive to build the model. - Needs more data to train, validate and testing. - Requires several weather variables as an input.
SARIMA	- Less time-intensive to build the model. - Needs less data for orders determination and testing. - Only requires I_g data as an input.	- Needs the previous five days data to make the imputation. - Has a slightly worse performance. - Each imputation takes around five minutes to be done.

Table 5.1: Advantages and disadvantages of models compared with the other.

temperature of the thermal zones (T_i) for both simulations are compared.

The metrics used to evaluate the impact of imputations are presented on Section 4.3. The behavior of the absolute error of T_i (T_i^{err}) between IS and BS is evaluated using three metrics. The vanishing time (V_t) is the metric that measures the time for the T_i^{err} to become negligible for the simulation, when $T_i^{err} < 0.1^oC$. The maximum temperature absolute error (T_{err}^{max}) measures the maximum T_i^{err} that is reached as a consequence of the imputation. T_{err}^{max} brings a magnitude of the error that can be expected when imputing with the CNN-LSTM model. The maximum error time (max_t) measures the time from when the imputation is started until T_{err}^{max} occurs. max_t brings us the location of time in which T_{err}^{max} is encountered.

Quantitatively, the metrics are shown with an average value of the eight imputed days. The largest averaged V_t has a value of 3.7 days and is encountered on N2AU202. While the smallest average V_t has a value of 1.8 days on PBCOORCOFI. The largest averaged T_{err}^{max} has a value of 2.7^oC on PBVESTIBULO. While the smallest averaged T_{err}^{max} has a value of 0.5^oC on PBATENCIONCOFI.

The values of V_t are the ones that shows the main advantages of this work. Commonly, when a day of data is missing the simulation needs around

15 days to reestablish itself. When this missing data is imputed using the CNN-LSTM it would take approximately a maximum of 3.7 days on this study case.

Additionally it was found a slight correlation between the daily T_i oscillation (T_{osc}) and T_{err}^{max} of thermal zones. This could help to estimate T_{err}^{max} when the measured data is not available.

Recommendations for future research works could be a more deepened analysis of the thermal behavior of the building. Also a replication of the methodology of Chapter 4, but now using SARIMA model imputations can be done. As well as the impact of imputations when time horizon of the models are increased for more than one day.

Finally, the Python scripts used for the development of this work can be encountered on the repository found in `https://github.com/alejgdr/meteorological_data_imputation`.

Bibliography

[1] Joseph L Schafer. Multiple imputation: a primer. *Statistical Methods in Medical Research*, 1999.

[2] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bolletin of mathematical biophysics*, 1943.

[3] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958.

[4] Marvin Minsky and Seymour Papert. *Perceptrons, An Introduction to Computational Geometry*. MIT Press, 1969.

[5] Geoffery E. Hinton, Terrence J. Sejnowski, and David H. Ackley. Boltzmann machines:constratint satisfaction networks that learn. Technical report, 1984.

[6] J.J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of National Academy of Sciences*, 1984.

[7] D. Rumelhart, G. Hinton, and R. Williams. Learning representations by back-propagating errors. *Nature*, 1986.

[8] S. J. Robinson and F. Fallside. The utility driven dynamic error propagation network. *Bolletin of mathematical biophysics*, 1987.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[10] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. 2006.

[11] R. Ahmed, V. Sreeram, Y. Mishra, and M.D. Arif. A review and evaluation of the state-of-the-art in pv solar power forecasting: Techniques and optimization. *Elsevier*, 2020.

[12] un Ma, Jack C.P. Cheng, Feifeng Jiang, Weiwei Chen, Mingzhu Wang, and Chong Zhai. A bi-directional missing data imputation scheme based on lstm and transfer learning for building energy data. *Energy & Buildings*, 2020.

[13] Alfredo Nespoli, Emanuele Ogliari, Sonia Leva, Alessandro Massi Pavan, Adel Mellit, Vanni Lughi, and Alberto Dolara. Day-ahead photovoltaic forecasting: A comparison of the most effective techniques. *Energies*, 2019.

[14] Lucas Borges Ferreira and Fernando França da Cunha. Multi-step ahead forecasting of daily reference evapotranspiration using deep learning. *Computers and Electronics in Agriculture*, 2020.

[15] Giorgio Guariso, Giuseppe Nunnari, and Matteo Sangiorgio. Multi-step solar irradiance forecasting and domain adaptation of deep neural networks. *Energies*, 2020).

[16] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. Statistical and machine learning forecasting methods: Concerns and ways forward. *Computers and Electronics in Agriculture*, 2020.

[17] Munir Husein and Il-Yop Chung. Day-ahead solar irradiance forecasting for microgrids using a long short-term memory recurrent neural network: A deep learning approach. *Energies*, 2019.

[18] Cyril Voyant, Gilles Notton, Soteris Kalogirou, Marie-Laure Nivet, Christophe Paoli, Fabrice Motte, and Alexis Fouilloy. Machine learning methods for solar radiation forecasting: A review. *Elsevier*, 2017.

[19] Munir Husein and Il-Yop Chung. Day-ahead solar irradiance forecasting for microgrids using a long short-term memory recurrent neural network: A deep learning approach. *Energies*, 2019.

[20] Muhammad Aslam, Jae-Myeong Lee, Hyung-Seung Kim, Seung-Jae Lee, and Sugwon Hong. Deep learning models for long-term solar radiation forecasting considering microgrid installation: A comparative study. *Energies*, 2019.

[21] Brahim Belmahdi1 and Mohamed Louzazni andvAbdelmajid El Bouardi1. A hybrid arima–ann method to forecast daily global solar radiation in three different cities in morocco. *The european physical journal plus*, 2020.

[22] Mario Tovar, Miguel Robles, and Felipe Rashid. Pv power prediction, using cnn-lstm hybrid neural network model. case of study: Temixco-morelos, méxico. *Energies*, 2020.

[23] Stefano A.BiniMD. Artificial intelligence, machine learning, deep learning, and cognitive computing: What do these terms mean and how will they impact health care? *The Journal of Arthoplasty*, 2018.

[24] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 2015.

[25] Pedro Ponce Cruz. *Inteligencia Artificial con aplicaciones a la ingenieria*. Alfaomega, 2010.

[26] Yu-Chen Wu and Jun wen Feng. Development and application of artificial neural network. *Wireless Pers Commun*, 2018.

[27] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow*. O'Reilly Media, 2019.

[28] Michael Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[29] Harini Suresh and Nicholas Locascio. *An introduction to LSTMs in TensorFlow*. MITCBMM ,`https://youtu.be/l4X-kZjl1gs`, 2018.

[30] K. Fukushima and S. Miyake. Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position. 1982.

[31] Y. LeCun, B. Boser, J.S. Denker, R.E. Howard, W. Habbard, and L.D. Jackel. Handwritten digit recognition with a back-propagation network. 1990.

[32] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J. Inman. , 1d convolutional neural networks and applications: A survey. 2021.

[33] Ratnadip Adhikari and R. K. Agrawal. An introductory study on time series modeling and forecasting. `https://arxiv.org/pdf/1302.6613.pdf`.

[34] R.J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice, 2nd edition*. OTexts, 2018, `https://otexts.com/fpp2/`. Accesed on 28 February 2022.

[35] Aric Labarr. *What is time series data?* `http://ariclabarr.com/youtube-timeseries.html`, 2021.

[36] Robert Nau. *Statistical forecasting: notes on regression and time series analysis*. Fuqua School of Business, Duke University, 2020.

[37] José de Jesús Quiñones Aguilar, Isaías Moreno Cruz, Camilo A. Arancibia Bulnes, Fernando U. Morales López, and David Riveros Rosas. Año meteorológico típico para Temixco, Morelos. 2016.