Universidad de La Laguna

Final Thesis

Image preprocessing techniques with application in nowcasting deep learning models

Double Degree of Physics and Mathematics

Author

Pérez Romero, Paula

Scientific driver

Prof. Dr. Guerrero Lemus, Ricardo

May, 2023

Abstract

The incorporation of photovoltaic energy into the world's electricity grid is already a fact. Therefore, forecasting the energy produced by photovoltaic plants is a topic that is increasingly being studied by the scientific community. The importance of developing good prediction techniques lies mainly in the intelligent and efficient use of this great resource, solar energy. With this motivation, this work will focus on short-term photovoltaic power forecasting techniques, known as nowcasting. The dataset we will use is composed of images from sky cameras and historical power data simultaneously obtained. This study will be focussed on the impact of different image pre-processing methods on the quality of the results in a deep learning model very similar to those used for nowcasting. Specifically, four techniques are proposed for preprocessing images before they are used to train a convolutional neural network (CNN). The results obtained in each of the four cases will be analysed in an exhaustive way with different metrics used for these purposes in order to make an informed comparison of them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The surface of Earth receives a total value of $3.85 \cdot 10^{24}$ J per year [1]. Consequently, in the words of Morton et al.[1], the solar energy received by the Earth every hour is enough to power the entire globe for a year. These data can help us to understand the power of the use of solar resources in world energy production.

The global cumulative photovoltaic capacity grew at an almost exponential rate since 2000 until 2021 as shown in Fig.1.1. It is expected to supply 16% of global electricity demand in 2050, according to the International Energy Agency (IEA)[2].

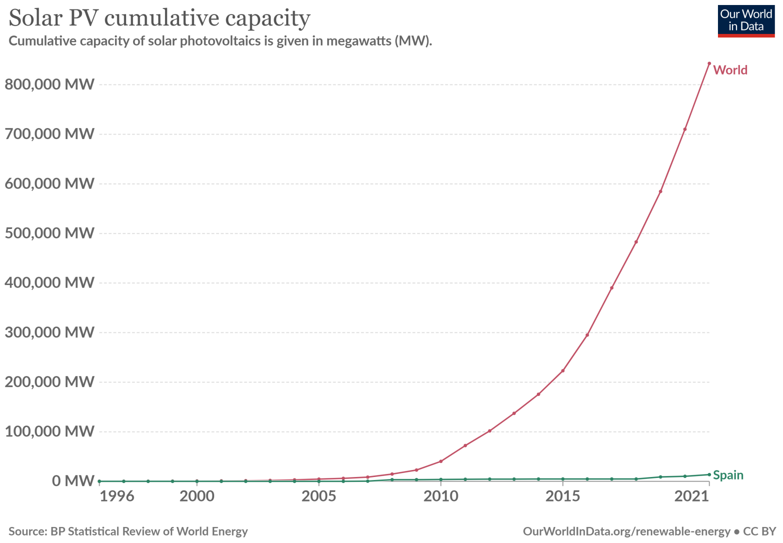


Figure 1.1: Graph of global cumulative photovoltaic capacity growth. Extracted from Our World in Data [3].

With the increasingly widespread adoption of solar photovoltaic plants around the world and its growing integration into the existing electricity grid, the forecasting of solar irradiance and photovoltaic (PV) power generation has become a topic of not only scientific, but also economic and strategic interest [4]. This is due to the fact that, as a renewable energy source, the solar energy production depends on external weather factors that cannot be controlled such as the position and movement of clouds. Therefore, forecasting the power output of solar photovoltaic panels is a critical factor in enhancing the efficiency of electricity network operating points in the presence of high solar energy penetration. In recent decades, this has become a widely studied topic by the scientific community, which has proposed many different solutions to this challenge.

The present work will focus on one of them, Skycam-based forecasting techniques, which are built on the use of Sky cameras, which are fisheye cameras capable of taking a picture of the entire sky dome. An example of an inexpensive ground-based skycam is shown below in 1.2. Traditionally, these devices have been prohibitively expensive, but with the advent of inexpensive digital security cameras and more powerful graphical processing capabilities, the wider adoption of localised solar forecasting it is much more practical than ever before [5].

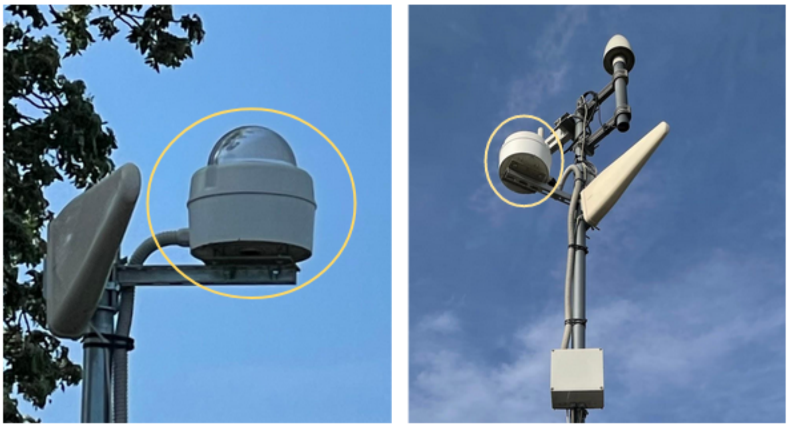


Figure 1.2: Two sky cameras located at the site where the data collection for this work was performed.

Skycam-based forecasting techniques have been proved to make important contributions to nowcasting and short-term solar forecasting, by identifying and predicting cloud movement and forecasting changes in solar availability. The term nowcasting should not be confused with the concept of short-term forecasting[1], as the former refers to the prediction of values that are occurring at the same time as the data used

[1] The definition of these terms will depend on the source consulted, as the issue is still under discussion. In this paper we will stick to the definitions given in this section. Generally, and in the context of this study, nowcasting techniques are used to predict weather conditions at the time of data collection. These predictions will be used to make new short-term predictions based on them.

to make the prediction, i.e. it is a simple regression problem [6], while the latter refers to the problem of predicting what's going to happen in the near future [7]. Not only nowcasting but also short-term forecasting are critical for managing smart grid operations such as system integration, ensuring power continuity, and managing ramp rates. In this context, in accordance with Zhang et al.[8] most numerical weather prediction (NWP) and satellite data are inadequate. This is due to their low spatial and temporal resolution, in other terms, the weather stations closest to the panel or PV plant being predicted are often too far away to provide reliable data, and on the other hand, satellite data often have relatively limited resolution.

With the aim of solving this issue, inexpensive ground-based sky imaging cameras (or 'skycams') installed close to the panels are a the best option, in accordance with Chow et al.[9]. Far from providing us with relevant weather information, the sky imagers provide us with images of the sky that will be analyzed to determine the relation between the sky appearance and the photovoltaic power output or the solar irradiance.

Nowadays, artificial neural networks (ANNs) are the tool of choice for analysing photos of the sky. It is worth mentioning that this has not always been the case. Other resources used for this purpose, which became more prominent in the past years, are optical flow techniques[2] and clear sky libraries[3] [10]. This work will focus on ANNs, in particular, in convolution neural networks (CNNs), which are the type of ANN most used with this aim as they can be trained to learn to distinguish details in an image, such as the area occupied by clouds in the sky dome or the position of the sun among others [11]. As reported by Saha [12] the CNN has an excellent performance in machine learning problems, specially the applications that deal with image data.

As mentioned above, this work will focus on the study of photo-

[2] Optical flow techniques are used in the forecasting of solar photovoltaic power generation to estimate the position and movement of clouds from image sequences. Optical flow refers to the pattern of pixel movement from one image to another, which can be used to estimate the speed and direction of cloud movement. With this information, changes in solar irradiance reaching the solar panels can be predicted more accurately.

[3] Clear sky libraries are used to predict the direct solar radiation reaching the solar panels on clear sky days. These libraries are based on models that use the position of the sun, time of day, latitude, altitude and other meteorological factors to calculate the amount of direct solar radiation reaching the solar panels.

voltaic power prediction techniques with sky cameras. It is necessary to clarify that it is not intended to develop any prediction technique, but concentrate on the previous step, the preprocessing of the images obtained with the sky-cams and its impact on the results of a deep-learning model whose architecture is similar to that of the mentioned models used for nowcasting (short-term forecasting). Image preprocessing is a critical step in using CNNs for computer vision. By performing transformations on images before feeding them to the network, the accuracy and computational efficiency of the neural network can be significantly improved. All these aspects will be addressed in this project.

Chapter 2

Objectives

With the motivation of studying this undoubtedly popular topic in numerous sectors, the objectives of this work have been raised. This work is based on the difficulty that deep learning models have in the task of detecting clouds and sun position in a sky image and therefore in predicting solar PV production and solar irradiance. As it was mentioned in the introduction, while this is a simple task for humans, it is actually quite complex for deep learning models to perform efficiently for all parts of the sky, with a variety of cloud types and for the variety of lighting and weather conditions throughout the day [5].

In this line, this work will compare the results of a machine learning model based on an artificial neural network applied in four different ways. The images that will serve as input data for the model will be pre-processed in each case in a specific form. Along these lines it will be possible to study the repercussion of each pre-processing method on the results of the model, results that can be extrapolated to more complex prediction models than the one used in this work. For this purpose, appropriate evaluation metrics, which will be discussed later, will be used.

In order to facilitate the understanding of future sections, a brief description of each of the four applications mentioned is presented below:

- In the first application of the model, which will be referred to as ”RGB model” or ”original model”, the input to the neural network we will be the RGB (red-green-blue) color images of the sky extracted from the skycams as (width, height, 3) arrays.
- In the second application of the model, which will be referred to as ”5-channels model” or ”complex model”, two additional channels

will be added to the three (RGB) channels of the original images, therefore, the inputs will be (width, height, 5) arrays. The first of the extra channels will be the nRBR (Normalized Red-Blue Ratio) channel constructed from the R (red) and B (blue) channels of the original image. The second channel will correspond to the saturation channel of the HSV (Hue, Saturation, Value) colour model image, the choice of which is explained below.

- In the third application, which will be referred to as "2-channels model" or "intermediate model", the input images will be (width, height, 2) arrays. The first channel shall correspond to the blue channel of the original image and the second to the nRBR channel calculated in the same way as in the 5-channels model.
- In the last application, which will be referred to as "nRBR model" or "simple model", the inputs shall be single channel images, (width, height, 1), corresponding to the nRBR channel calculated in the same way as for the 2-channels and 5-channels models.

The decision to choose the nRBR and the saturation channels to combine with the red, green and blue channels of the original image was not arbitrary. Numerous scientific studies have shown the effectiveness of these image pre-processing methods in detecting clouds in sky images [13, 14, 15, 16, 17, 4], etc.

On the one hand, the use of the saturation (S) channel of an image in the HSV colour space for solar photovoltaic prediction can be interesting for several reasons: The saturation channel is a measure of the purity of the colour in an image. Higher saturation indicates higher colour purity and, in the context of PV power prediction, this may be correlated with higher intensity of incident sunlight. Therefore, the saturation channel can provide useful information on the amount of solar radiation reaching the solar panels. Moreover, compared to other channels, such as the hue channel (H) or the value channel (V), the saturation channel is less susceptible to variations in illumination. This means that even in changing lighting conditions, the saturation channel can provide useful and consistent information about the purity of colour in an image. For all these reasons, the HSV colour space is commonly used in image processing and computer vision because of its simplicity and efficiency in calculating colour transformations. In addition, many machine learning algorithms, including convolutional neural networks (CNNs), can process images in the HSV colour space effectively [4, 17].

On the other hand, talking about the concept of the Red-Blue Ratio for cloud algorithms, it is worth mentioning that it was first developed at Scripps Institution of Oceanography with the Whole Sky Imager (WSI) [13].

A sample of the red-blue ratio applied to a clear sky image and to a cloudy day image extracted from Long et al.'s paper "Retrieving Cloud Characteristics from Ground-Based Daytime Color All-Sky Images", specifically from Figure 4, on page 638, is shown in Fig.2.1.

Below the sample sky image are two images that show the corresponding extracted red–green–blue (RGB) color channel blue and red pixel values that make up the sample image. The red pixel values are relatively small (dark) in the sky portion of the image because little red light is scattered by this clear atmosphere compared to the correspondingly greater blue scattering and greater blue pixel values, except near the horizon where the increased atmospheric pathlength makes the original sky image appear white to our eyes, and somewhat near the sun in the image. The corresponding relative red/blue ratio values are shown in the upper second from left image in Fig.2.1. For clear sky the red/blue ratio is small, that is, dark in the image, but increasing near the sun and near the horizon.

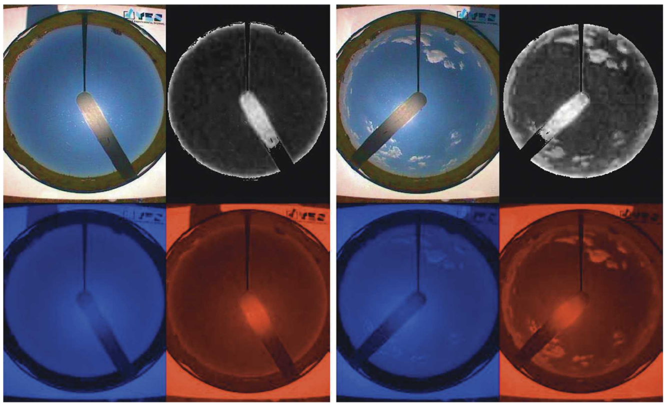


Figure 2.1: (top left) Clear-sky image obtained by a skycam, (top second from left) corresponding relative red/blue ratio “image,” (lower left) separated blue, and (lower second from left) red pixel value amount images. (top third from left) Cloudy-sky image, (top right) corresponding relative red/blue ratio “image,” (lower third from left) separated blue, and (lower right) red pixel value amount images [13].

The purpose and basis of this study lies in the following idea: The image preprocessing can determine such decisive factors when assessing the performance of a deep learning model as the quality of its results and the computational cost required for training, among others. If the results were satisfactory, there would be no doubt about the potential for the application of these methods in more complex solar forecasting models.

Chapter 3

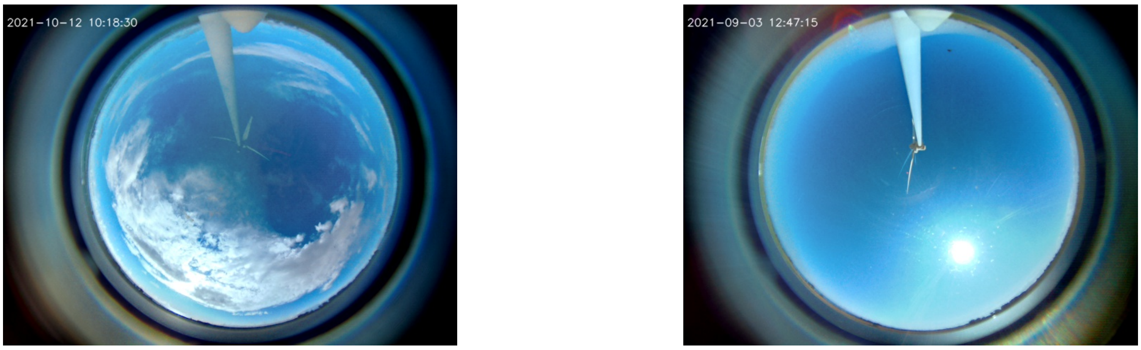
Methodology

3.1 Data

3.1.1 Data capture

3.1.1.1 Images

The original dataset consists of a set of images of the sky obtained at a frequency of 15 seconds using a fisheye (also known as wide-angle) sky-camera capable of photographing the sky with a field of view of up to 180 degrees. The images were obtained in Simrishamm, Sweden, near a photovoltaic plant. All the images were organized by date and time of recording.



(a) Cloudy day.

(b) Clear day.

Figure 3.1: Examples of sky images obtained with the skycam on different days.

3.1.1.2 Historical active power data

The second part of the dataset corresponds to active power data obtained every 10 seconds from the PV plant near the sky-cam location. The corresponding power data were organised by date and time of recording in a csv document.

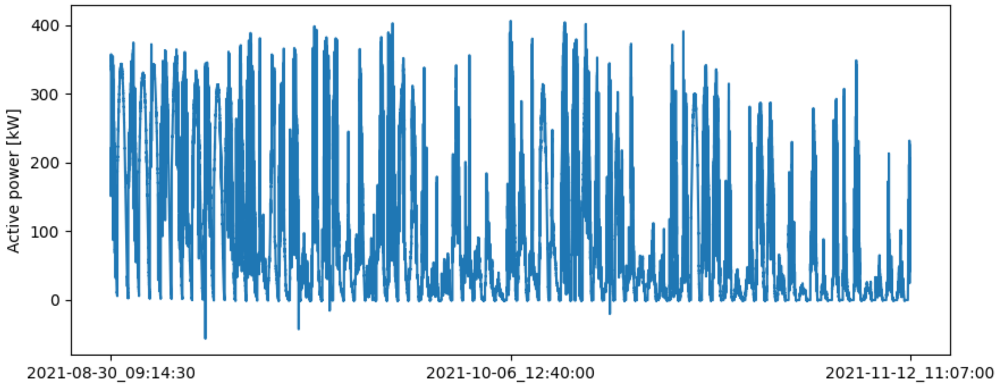


Figure 3.2: Active power dataset.

3.1.2 Data preprocessing

Preprocessing of the dataset, both power data and images, is a crucial process that must be performed before training any model.

3.1.2.1 Interpolation of the active power data

Since the power data were obtained at a frequency of 10s while the images were obtained every 15s, it was necessary to interpolate the power data in order to have it with a frequency of 15s, so that each image had its corresponding power data.

3.1.2.2 Selection of the data

Since a very large dataset was available, it was possible to make a selection. The dataset that was finally used to train the model was composed of 185,808 photos and their corresponding power data recorded every 15 seconds from 9:00 a.m. to 19:30 p.m. thus eliminating the dark images corresponding to the hours of less light. The selected images correspond to those obtained from August 30 to November 12, 2021.

3.1.2.3 Dataset division into training, test and validation data

The selected dataset will be divided into three subsets of data: training, validation and test. Each of these subsets will have a particular size and function in relation to the model. While the training set will be used to adjust the hyperparameters of the neural network, the validation set will be used to evaluate the model at each epoch or training step. Finally, the test set will be used to make the final evaluation of the model once all the training steps have been completed. In this work, 75% of the data has been selected for training, and of the remaining 25%, the 25% will be used for validation, leaving 18.75% of the original dataset for the test set.

3.1.2.4 Normalization of the power data

Data normalization is performed to speed up the learning process of the neural networks. We ensure that different features have similar ranges of values (feature scaling) to enable the gradient descent to converge faster. Therefore, we normalize the training data to solve the model learning challenge. For normalizing the power data the the MinMaxScaler from Sklearn (is one of the most popular machine learning libraries in Python) whose formula is 3.1 will be used. The resulting data will be in the range [0, 1].

$$X_f = \frac{X_0 - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

The normalization of the images is performed in a different way. It will be detailed in the image preprocessing subsection.

3.1.2.5 Image preprocessing

In order to reduce the image noise and to select the part of the images corresponding to the sky circle, a mask was applied as shown in the 3.3.

In addition, all images were resized from 640x480 to 100x100 pixels. Image resizing is a process that must be performed before loading the images into the model [18]. Its importance lies mainly in the following reasons: Firstly, CNNs can require large amounts of memory and processing time, especially if the images are very large. By resizing the images to a suitable size, the processing time and memory usage of the neural network can be reduced, which can allow for faster and more efficient training of the model. Secondly, the input images must have the same size to be processed in a CNN model. If the images have different sizes, they can introduce noise into the model and hinder the neural network's ability to learn useful patterns in the images. By resizing all input images to the same size, you can normalise the size and ensure that all images are processed in the same way. Finally, by resizing the images, you can increase the chances that the neural network will learn more general patterns that can be applied to a variety of images, not just training images, thus improving its ability to generalise to new images.

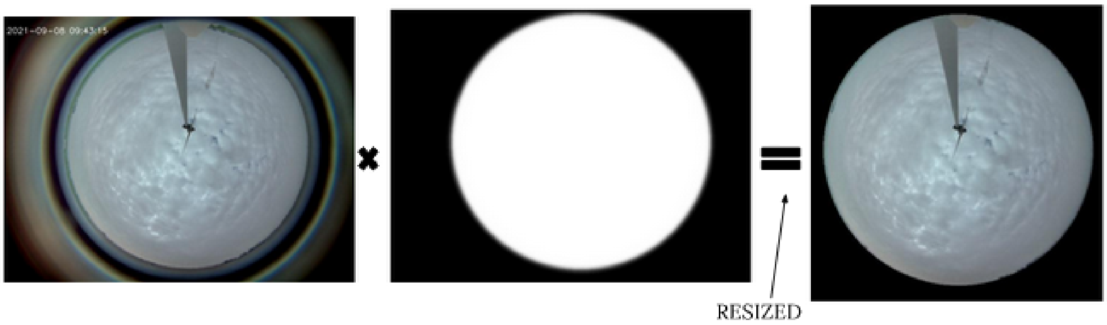


Figure 3.3: Applying the mask and resizing to the original image.

Once the images were resized, their preprocessing was differentiated according to the model.

- RGB model
  In this case no extra preprocessing is performed, as the original images are in RGB color format.

- 5-channels model
  The nRBR channel is calculated from the red and blue channels of the original image with the Eq.3.2 where R corresponds to the intensity of the red channel pixels and B to the blue channel pixels. The factor e in the denominator is essential to avoid division by zero errors.

  $$X_{nRBR} = \frac{R - B}{R + B + e} \tag{3.2}$$

  Subsequently, the original RGB image is transformed into an image in HSV color format and the saturation channel is extracted. Finally, these two new channels are added to the three channels of the original image.

- 2-channels model
  In this case, the calculation of the nRBR channel and the extraction of the saturation channel were performed in the same way as in the 5-channels model. In this case, the resulting image will consist of these two channels.

- nRBR model
  In this model, the nRBR channel was calculated in the same way

as for the 5-channel and 2-channel models. The resulting images consist of this single channel.

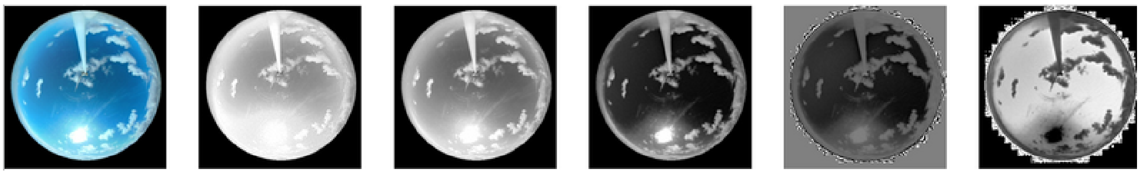


Figure 3.4: This image shows one of the images of the dataset corresponding to a sunny day with scattered clouds and the five channels (in grayscale) of the image resulting from it following the preprocessing described above. From left to right: original image RGB, blue channel, green channel, red channel, nRBR channel and saturation channel.

In all cases, the resulting images were normalized by dividing their pixel intensity by the maximum value as shown in Eq.3.3. This technique is also known as range normalization and is used to adjust the image pixel values in a range between 0 and 1.

$$X_{Normalized} = \frac{X}{255.0} \tag{3.3}$$

Range normalization is useful to ensure that the pixel values of different images are in the same range, which makes it easier to compare the characteristics of different images.

3.1.2.6 Batch size

In deep learning, a batch refers to a set of data points that is processed simultaneously during model training. Instead of feeding the entire training data set at once, the set is divided into several smaller batches and the model is iteratively trained on each batch.

The use of batches is common in deep learning due to the memory limitations of GPUs and CPUs, as well as to improve training efficiency. By processing data in batches, the model only needs to load a small amount of data into memory at a time, allowing training to be performed faster and in parallel.

Batch size is an important hyperparameter that can be adjusted during model training. A larger batch size may improve training efficiency, as the model takes less time to process the data, but it may also require more memory and may make the training more susceptible to overfitting. A smaller batch size may require more training time, but may improve generalization and avoid overfitting.

A batch size of 200 data has been chosen for this project. This decision was conditioned by the size of the dataset, the capacity of the computer processor used to train the model and by the complexity of the model.

3.2 Model

The model that will be used in this work and that will be used to draw conclusions about the different preprocessing techniques mentioned (see Sec.3.1.2.5) will be a convolutional neural network (CNN). CNNs are a type of deep learning algorithm that is mostly used to analyze and learn visual attributes from large amounts of data [19]. Although they are mainly used for image-related AI applications, they can also be used for other AI tasks such as natural language processing, as well as in recommendation engines[1].

In order to try to understand the functioning of this type of neural networks, it is necessary to visualise the images as arrays of dimensions width x height x depth (see Fig.3.5) where width and height are integers corresponding to the number of pixels in each of these dimensions and depth refers to the number of channels in the image.

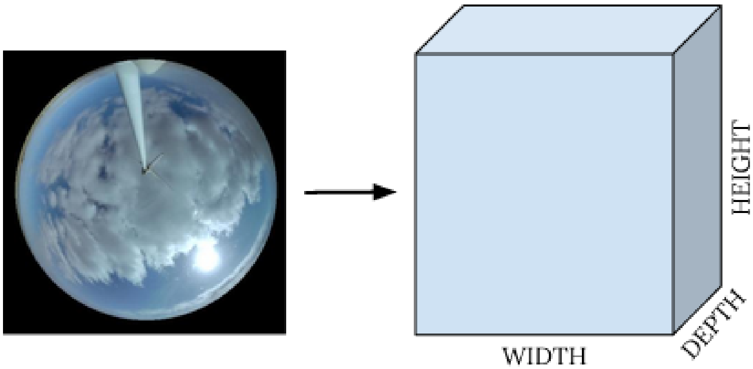


Figure 3.5: Image viewed as a matrix.

Taking this into account and trying to contextualise the usefulness of this type of network, the following situation is considered: if we wanted to connect a 100x100xC image to a neuron in our neural network, this would imply a total of 10000xC parameters for a single neuron. Herein lies the great usefulness of CNNs, as they are able to reduce the number of parameters to achieve greater efficiency.

[1] Recommendation engines are software systems designed to suggest products, services, content or other relevant information to users based on their preferences, interests or past behavior [20].

3.2.1 Model Framework

The architecture of this model was decided based on the capacity of the computer processor on which the training process was to be performed, on other examples of convolutional networks with applications related to object detection in images and on empirical tests performed comparing different architectures.

The model consists of four convolution blocks (see Fig.3.6), followed by six more layers. These last layers are, in order: a flatten layer, a dense layer with 128 neurons, a batch normalization layer, a rectified linear unit (ReLU) activation layer, a dropout layer and finally a dense mononeuronal layer. The input size for the convolutional network will be 200x100x100xC, where 200 corresponds to the batch size, 100x100 to the width and height of the images respectively and C corresponds to the number of channels of the images according to the received preprocessing.

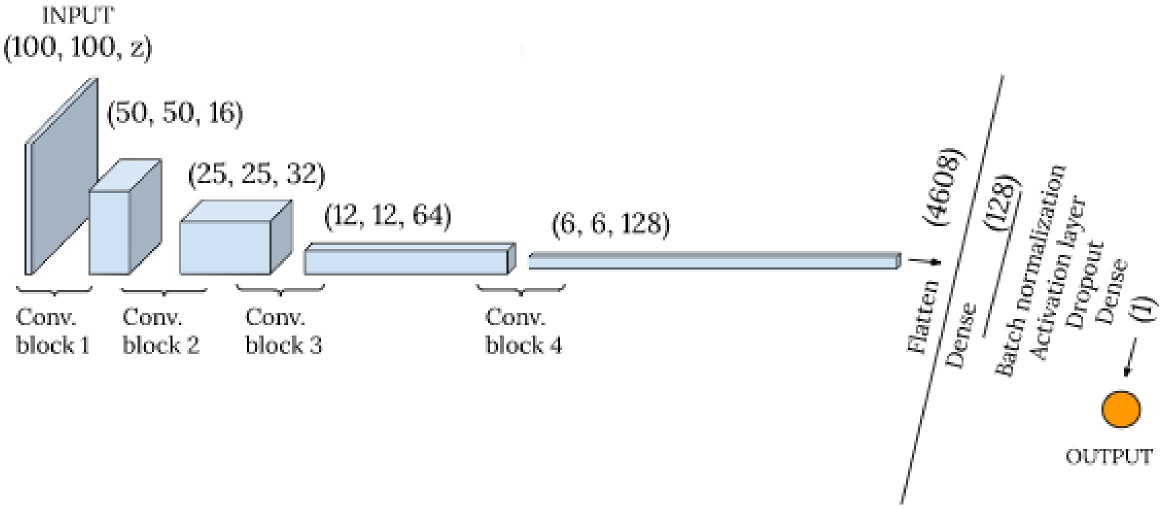


Figure 3.6: Diagram of the model used.The brackets specify the dimensions of the array at each stage of the process. The value of z shall be 1, 2, 3 or 5 depending on the nRBR, 2-channel, RGB or 5-channel model, respectively.

Each convolution block consists of a 2D convolution layer, a batch normalization layer, a ReLU activation layer and a 2D MaxPooling layer. Each of the above-mentioned layers will be explained below.

3.2.1.1 Convolutional layers

In convolution layers, a set of kernel filters are applied to the input image, which is treated as a matrix, and shifted over it (see Fig.3.7).

These filters, are themselves matrices of the same depth as the original image. While the number of filters in each convolution layer is determined, the filter values, which correspond to the weights of the trained CNN, are learned by the neural network during its training. The resulting filters will be those that have proven to be most effective on the assigned task.

The result of the convolution is a set of feature maps that highlight different patterns in the input image. The set will consist of as many matrices as the number of kernel filters applied during convolution, i.e. the depth of the resulting "image" will be equal to the number of filters.

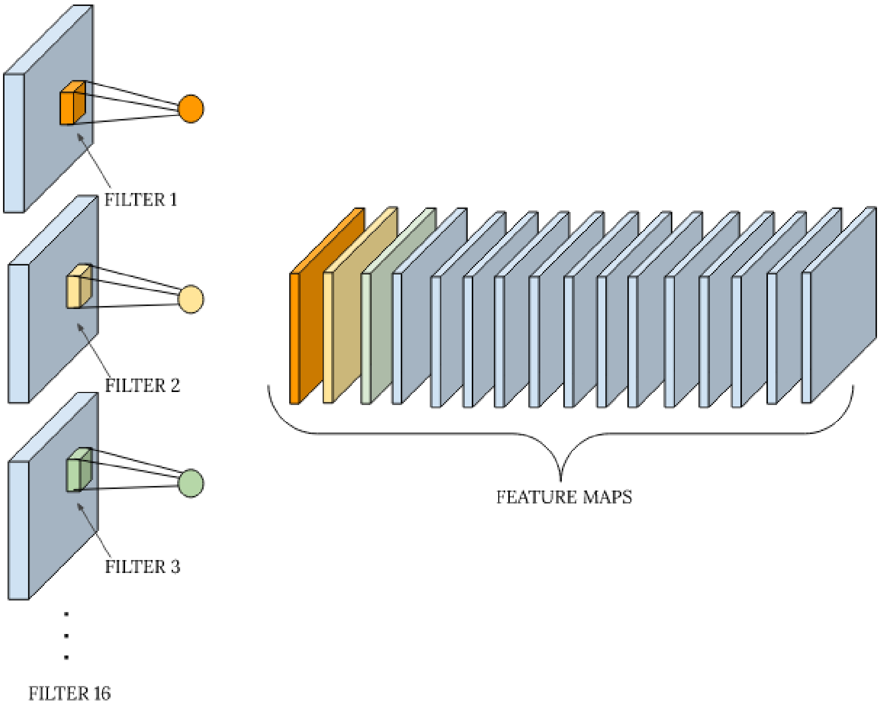


Figure 3.7: Representation of a 16-filter convolution process.

Figure 3.8 shows the output (set of feature maps) of a 2D convolution layer in which 16 filters of kernel 3x3 were used on a 100x100 resolution image.

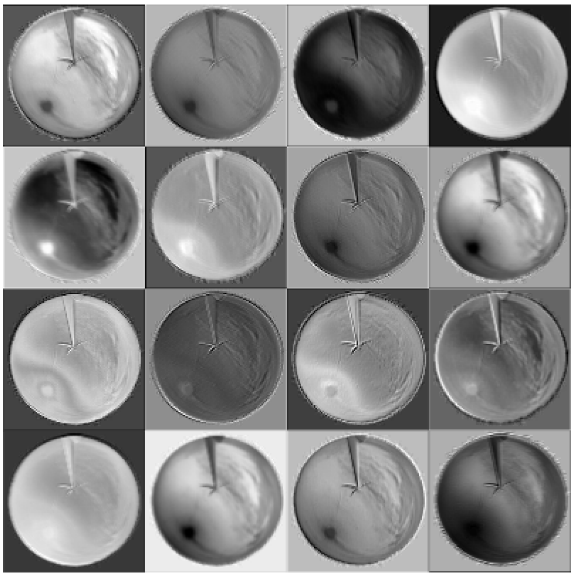


Figure 3.8: Set of feature maps obtained after the first convolution (with 16 filters) of the 5-channels model.

3.2.1.2 Batch normalization layers

Batch normalization layers are a popular regularization technique in deep learning. These layers are used to normalize the input from a previous layer, i.e., adjust the mean and variance of the input to a standard distribution. According to Ioffe et al. in [21], batch normalization layers have several benefits in the neural network, including:

- Reducing the covariate problem: Batch normalization layers reduce the covariate problem, which occurs when two features are highly correlated and, therefore, one can predict the other. This reduces redundancy in the neural network, which can improve its generalization ability.
- Training acceleration: Batch normalization layers can accelerate the training of a neural network. This is because normalizing the input of a previous layer reduces the variance of the activations of the current layer, which reduces the magnitude of gradients during training and thus accelerates convergence.
- Regularization: Batch normalization layers can also act as regularizers, which helps reduce overfitting in a neural network.

However, there are also some limitations in the use of batch normalization layers. For example, they may require more training time and higher memory consumption. In addition, they can also reduce the network's ability to capture complex patterns in the data if they are overused or misused.

3.2.1.3 Pooling layers

The main purpose of these layers is to reduce the dimensionality of the features generated by the previous convolutional layers, which helps to reduce computational cost and improve generalization. Pooling layers work by taking a region of the input image and applying a statistical operation to summarize that region into a single value. The most common operations used in pooling layers are maximum and average, which take the maximum or average value of the pixels within the region, respectively (see Fig.3.9). The operation is repeated with a window that scrolls through the entire input feature, generating a smaller output feature. [22].

However, while pooling layers can be useful for reducing the size of the input, they also have some limitations. For example, the use of these layers can lose spatial information, which can be critical for applications such as image segmentation. In addition, pooling can decrease spatial resolution and thus limit the ability of the network to capture fine details in the image.

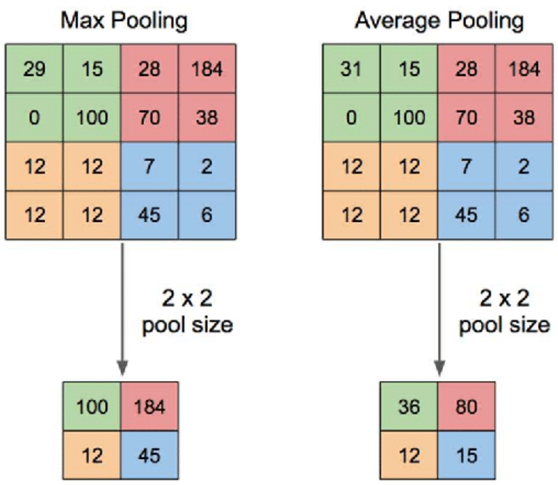


Figure 3.9: Max pooling vs. Average pooling scheme. Image extracted from [23].

In the model used in this work, it was decided to use max pooling instead of average pooling, based on the idea of Scheller et al. at [24], who state that this type of pooling has a higher convergence rate and better generalization capacity than average pooling.

3.2.1.4 Dense layers and activation function

In dense layers, each neuron is connected to all neurons in the previous layer. These layers are commonly used in neural networks to learn complex representations of input data. Each neuron in this layer receives as input a linear combination of the outputs of all neurons in the previous layer, followed by an activation function. This linear combination is realized by the dot product between the weights, w_i of the connections and the outputs of the neurons of the previous layer, x_i, to which a bias term [2], b, is added, (see Eq.3.4). The number of dense layer units is a fundamental parameter, as it determines the size of the output vector.

$$z = b + \sum_i w_i x_i \tag{3.4}$$

On the other hand, the activation function is a function used for transforming the input values of neurons. Essentially, it brings nonlinearity into the neural networks so that they can learn the relationship between the input and output values. [22].

In the model used, two dense layers were incorporated after the convolution blocks, one of 128 units and rectified linear function (ReLU) activation (see Fig.3.10) and the last layer, which will be another dense layer of a single unit and linear activation function.

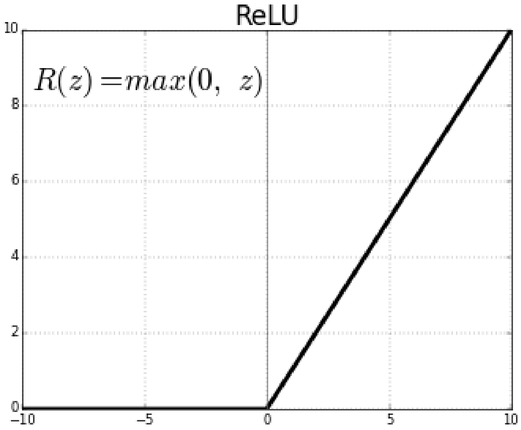


Figure 3.10: Activation function ReLU.

In a regression problem, such as the one covered in this paper, the last layer of the neural network is used to predict a continuous nu-

[2] The bias term is added to the linear combination of the inputs and connection weights before passing them through the nonlinear activation function. It is used to shift the activation function up or down along the y-axis. This means that the bias term affects the activation level of the neuron, independent of the inputs.

merical value rather than a discrete class. Therefore, it is important to choose the activation function and the number of neurons in this layer to ensure that the network can learn to produce a continuous and accurate output. This is why the linear function has been chosen, as it takes as input the linear combination of the inputs and the weights of the connections in the previous layer. Therefore, the output of the last layer is simply the weighted sum of the outputs of the previous layer, which allows the network to produce a continuous output without compressing the range of the output. Furthermore, with a single neuron, the network can produce a single output for the desired regression value, which in this particular case will correspond to an active power value.

3.2.1.5 Dropout layers

Dropout layers are an efficient regularization technique that helps prevent overfitting and improves model generalization. The dropout technique consists of randomly selecting a subset of neurons in a layer and temporarily disabling them during training. This prevents certain neurons from becoming overly specialized to specific patterns in the training data and forces all neurons in the layer to learn to be useful for the task at hand. After training, the neurons that were temporarily deactivated are activated again for model evaluation [25].

In the model used, a dropout layer is applied, where the 20% of the neurons are deactivated, before the last dense layer.

3.2.1.6 Flatten layers

Flatten layers convert a multidimensional tensor into a one-dimensional vector, allowing the output of a convolutional layer to become an input to a fully dense layer, since these layers only accept one-dimensional inputs.

For example, starting from the output of the last convolution block of the 5-channel model (see Fig.3.6) used, each image would have dimensions 6x6x128. By applying a flattening layer, this tensor can be converted into a one-dimensional vector of size $6 \cdot 6 \cdot 128 = 4,608$, which can be used as input for a dense layer.

3.2.2 Metrics

The metrics used to assess the performance of a CNN model during training and validation (such as training loss and validation loss) are different from the metrics used to assess the quality of model predictions on test data.

3.2.2.1 Metrics used to evaluate the performance of a model

Training and validation metrics measure the model prediction error on training and validation data, respectively. These metrics are used during the training process to adjust model parameters and improve model performance on training and validation data. The metrics used for this purpose were the mean absolute error (MAE) as the loss function and the mean square error (MSE).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3.5}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.6}$$

In addition, the training and validation time has also been considered as an indicator of the computational cost of each of the models. This metric is of great importance when designing a network architecture, especially when working with large amounts of data.

3.2.2.2 Metrics used to assess the quality of the model predictions

The metrics used to assess the quality of model predictions on test data measure the performance of the model on a separate dataset that has not been used during training or validation. These metrics indicate how well the model generalises to completely new data.

Following the recommendations of Boursalie et al. [26] , the coefficient of determination (R^2), the MAE and the root mean square error (RMSE), were used.

$$RMSE = \sqrt{MSE} \tag{3.7}$$

$$R^2 = 1 - \frac{MSE_{Model}}{MSE_{Ref}} \tag{3.8}$$

where $MSE_{ref} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i)^2$, using the mean of the inputs as reference.

R^2 is expected to take values between [0, 1], but it may happen, in cases where the model to be evaluated obtains worse predictions than the model that uses the mean as prediction, that this coefficient is negative. This coefficient determines the quality of the model to replicate the results, and the proportion of variation in the results that can be explained by the model. $R^2 = 1$ would indicate that the results can be explained by the model in their entirety.

Chapter 4

Results

4.1 Comparison of performance metrics

Using the metrics presented in section 3.2.2.1, the Figure 4.1 has been constructed. It can be seen that in all models the training loss is inversely proportional to the training time, and consequently, to the computational cost. The high training time of the 5-channels model compared to the rest of the models stands out. An increase of about 3 minutes is observed from the nRBR model to the RGB model, while from the RGB to the 5-channels model an increase of about 7.5 minutes is observed, having in both cases two channels of difference between the input images of the model.

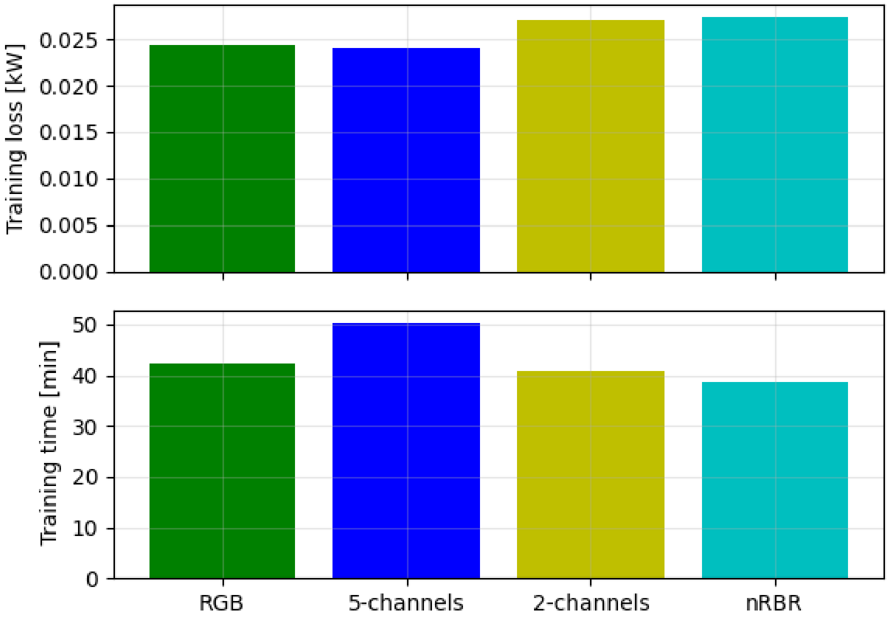


Figure 4.1: Comparison of training loss and training time of all models. See Table A.1.

4.2 Comparison of prediction metrics

As mentioned in section 3.2.2.2, the closer the value of R^2 is to the unity, the better the predictions. Taking this into consideration, it can be seen in the linear regressions of Fig.4.2, that all the models manage to explain more than 97% of the results, indicating that the architecture of the model designed is adequate for this purpose. Even though the differences are not very large, the one with the best predictions is the 5-channels model, followed by the RGB, 2-channels and the nRBR in last place. This was the expected result, the more channels the input image has, the more information is being offered to the model to better learn how to make predictions.

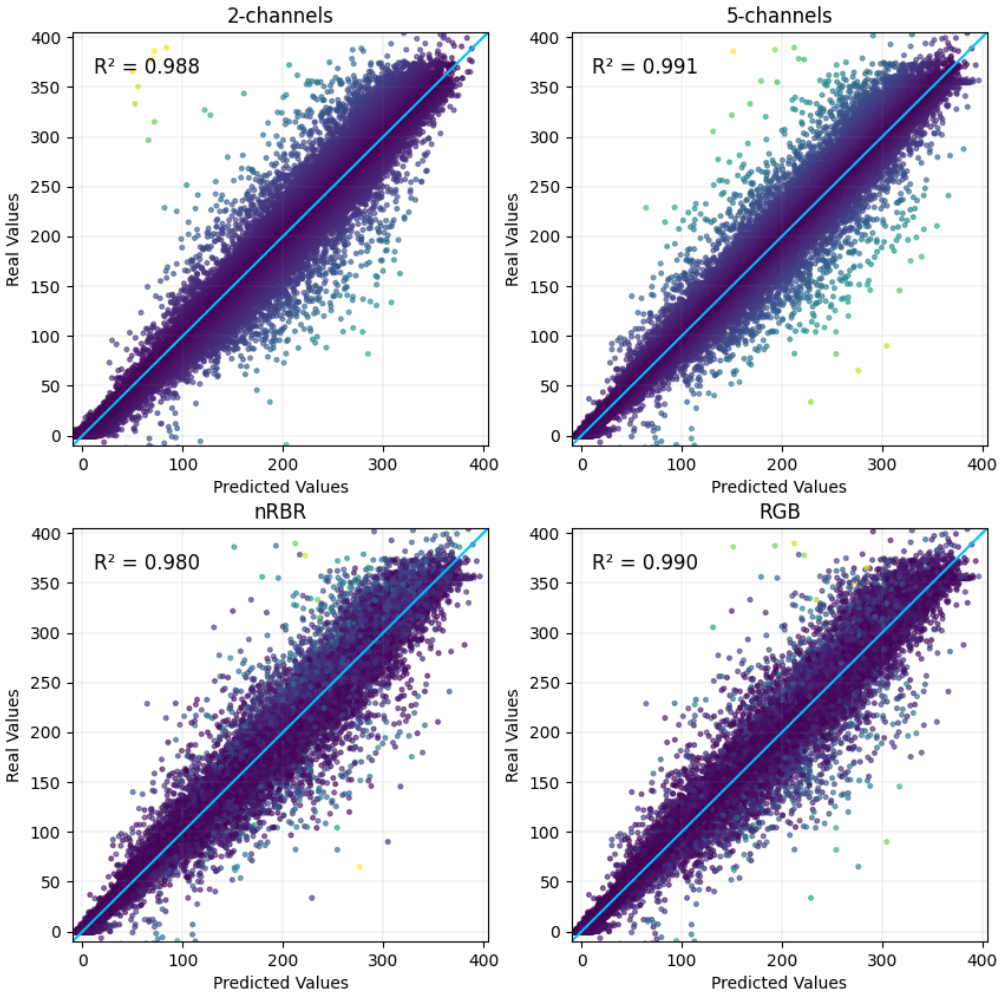


Figure 4.2: Linear regression between real and predicted data for all models. See Table A.2.

To reaffirm the validity of the models used, their error histograms have been plotted in Fig.4.3. According to Gawlikowski et al.[27], these plots are very useful to understand how the model is performing and how the errors in the predictions are being distributed. This can help to identify patterns in the errors and to find areas where the model needs improvement. For example, if the errors are normally distributed around zero, it may be an indicator that the model is making good predictions. On the other hand, if there are a large number of very large or very small errors, it may be an indicator that the model is having difficulty learning certain patterns in the data.

Although it can be seen in Fig.4.3 that, broadly speaking, all the plots resemble a normal distribution, there is a clear shift of the maximum frequency error to the right of zero in all of them. This means that there is a tendency in the model to underestimate the values of the target variable. In other words, the model is producing values that are below the true values. This could be due to several factors, such as imbalance in the distribution of the training data, a lack of representation of certain classes or features in the training data, or a lack of ability of the model to capture certain patterns in the data. Particularly, in the graph of the 5-channels model, this shift is minimal, which again shows the good performance of this model.

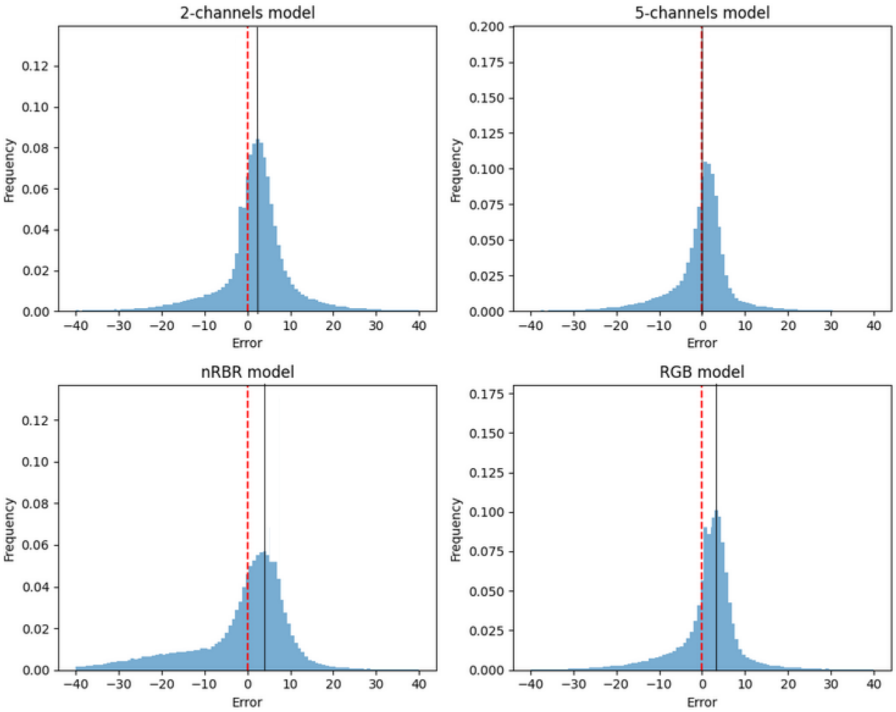


Figure 4.3: Error histograms of the four models. The black line indicates the error with maximum frequency.

In addition, the MAE and RMSE have been plotted for each of the models (see Fig.4.4). Reasonable values have been obtained for these metrics according to the results obtained by Zhang et al.[8]. Again, the more channels the input image has, the smaller the errors made by the model in making the prediction.

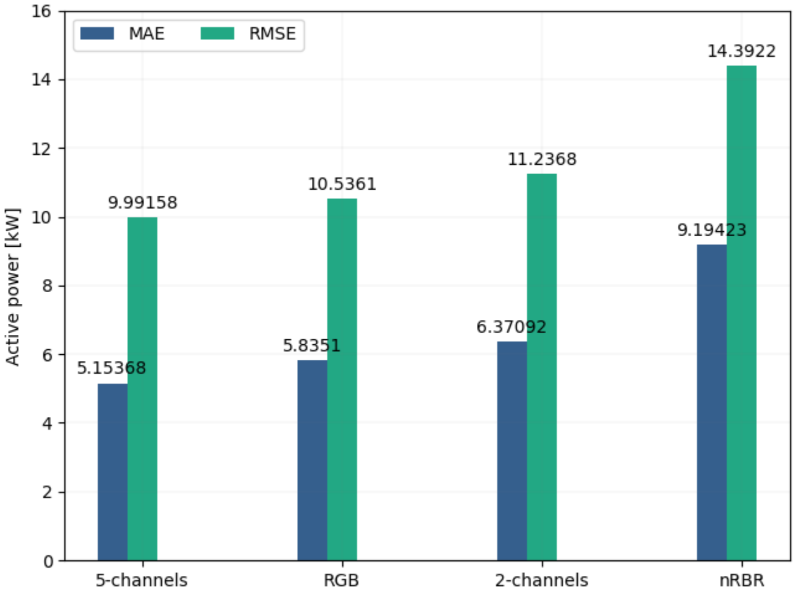


Figure 4.4: Comparison of MAE and RMSE of all models. See Table A.2.

In order to further analyse the results of each model, the daily forecasts have been studied. As can be seen in Fig.4.5, the days of maximum and minimum mean prediction error for the 5-channels model and the RGB model match. This result raises the question of the usefulness of the extra information provided by these two channels added to the RGB image in the 5-channel model.

Thanks to Fig.4.5, a uniform behaviour of all models has been observed. The days of minimum error coincide with days of cloudy skies and the days of maximum error coincide with days when there are changes in the state of the sky, e.g. days when it is sunny at dawn and cloudy as the day progresses, and vice versa. This result coincides with that of other nowcasting papers [5, 8, 4], where it is stated that the highest photovoltaic prediction errors are produced temporally coinciding with ramps in production, and these ramps are mainly due to a change in cloudiness.

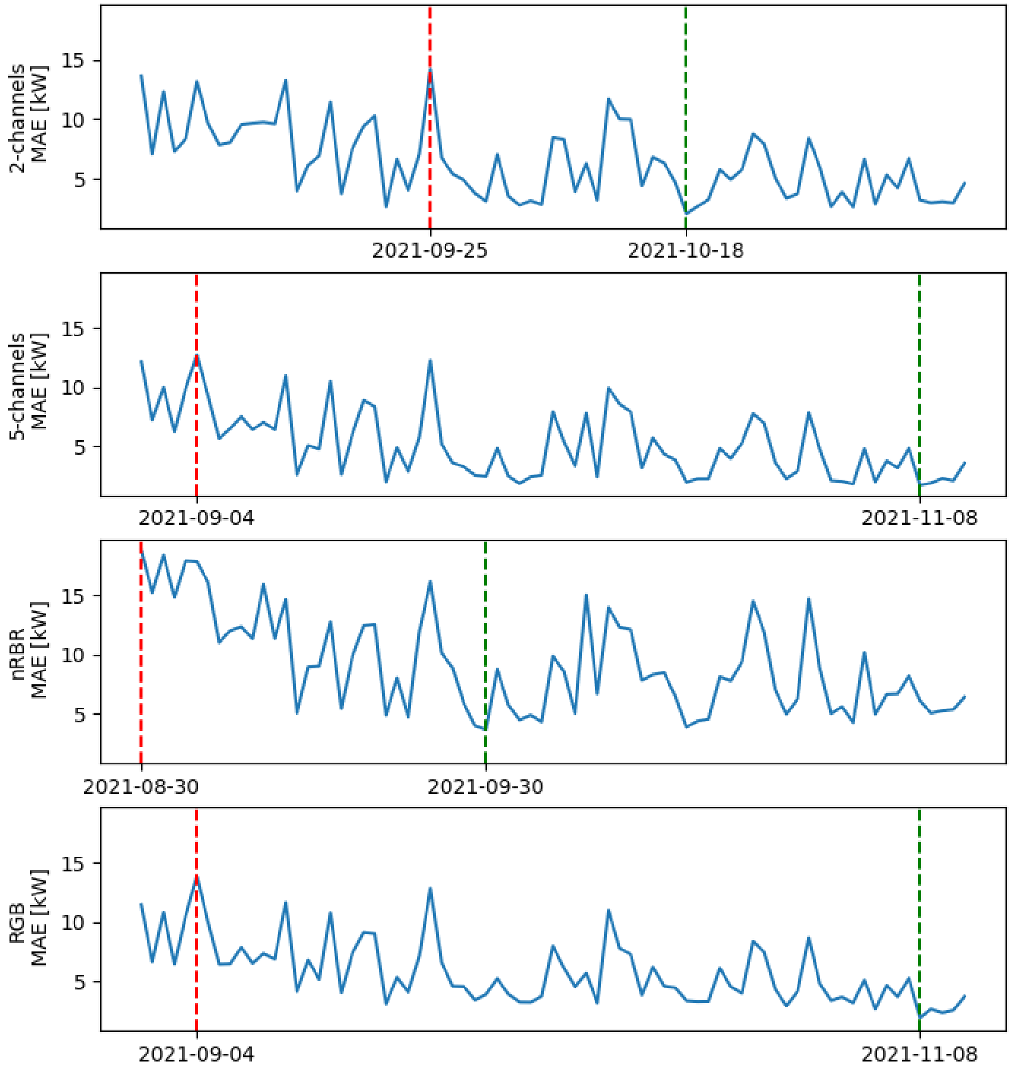
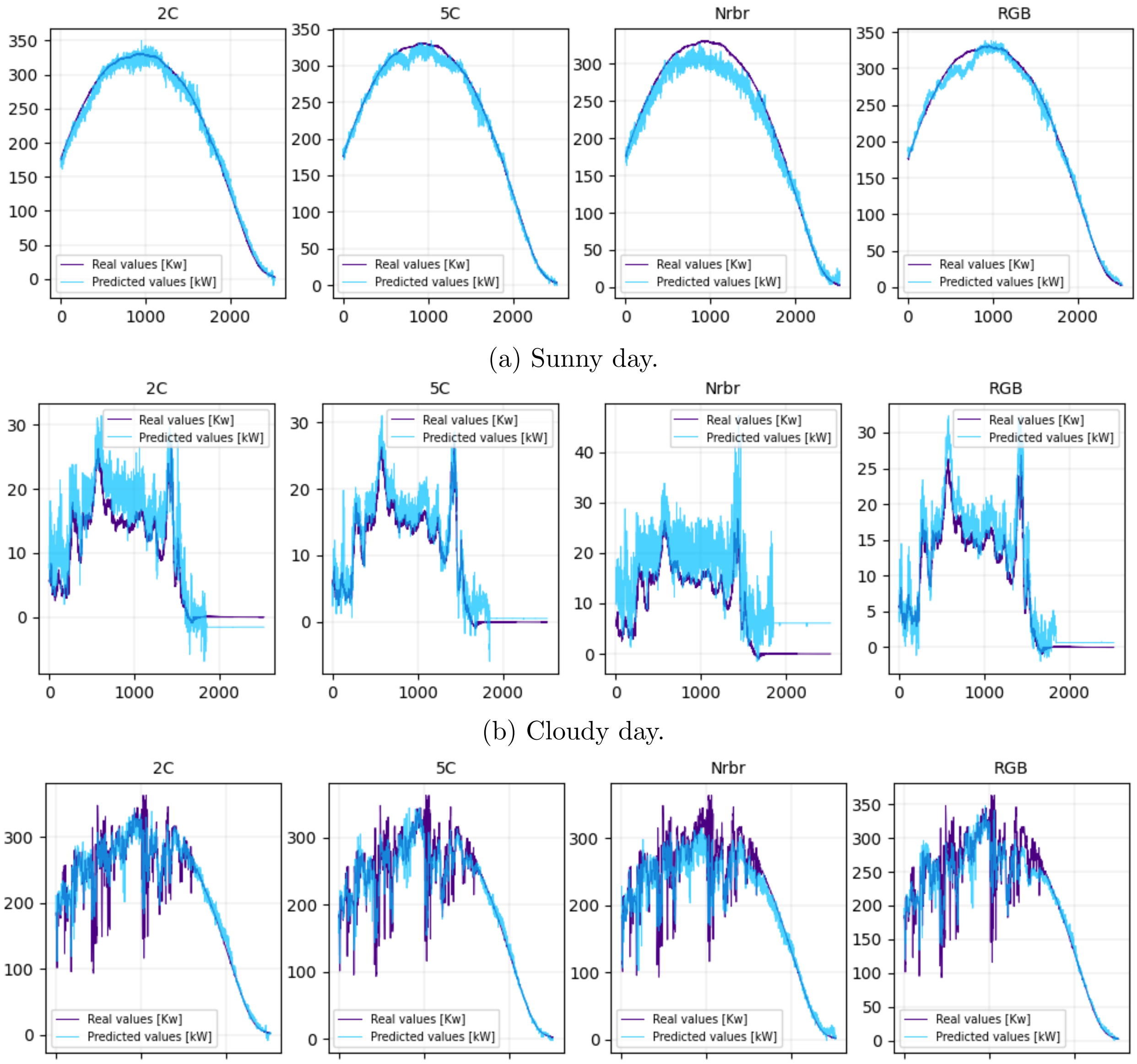


Figure 4.5: Comparison of the mean absolute error per day. See Table A.3.

To deepen this issue and to extract more useful information on the model results, three specific days have been chosen from the dataset: a sunny day, a cloudy day and a partially cloudy day. The graphs in Fig.4.6 show the performance of each model on each of the days mentioned.



(a) Sunny day.

(b) Cloudy day.

(c) Partially cloudy day.

Figure 4.6: Values predicted by the different models versus actual values for a sunny, cloudy and partly cloudy day.

The R^2 has been calculated for each of these days, and for each of the models. The resulting plots can be seen in Fig.4.7.

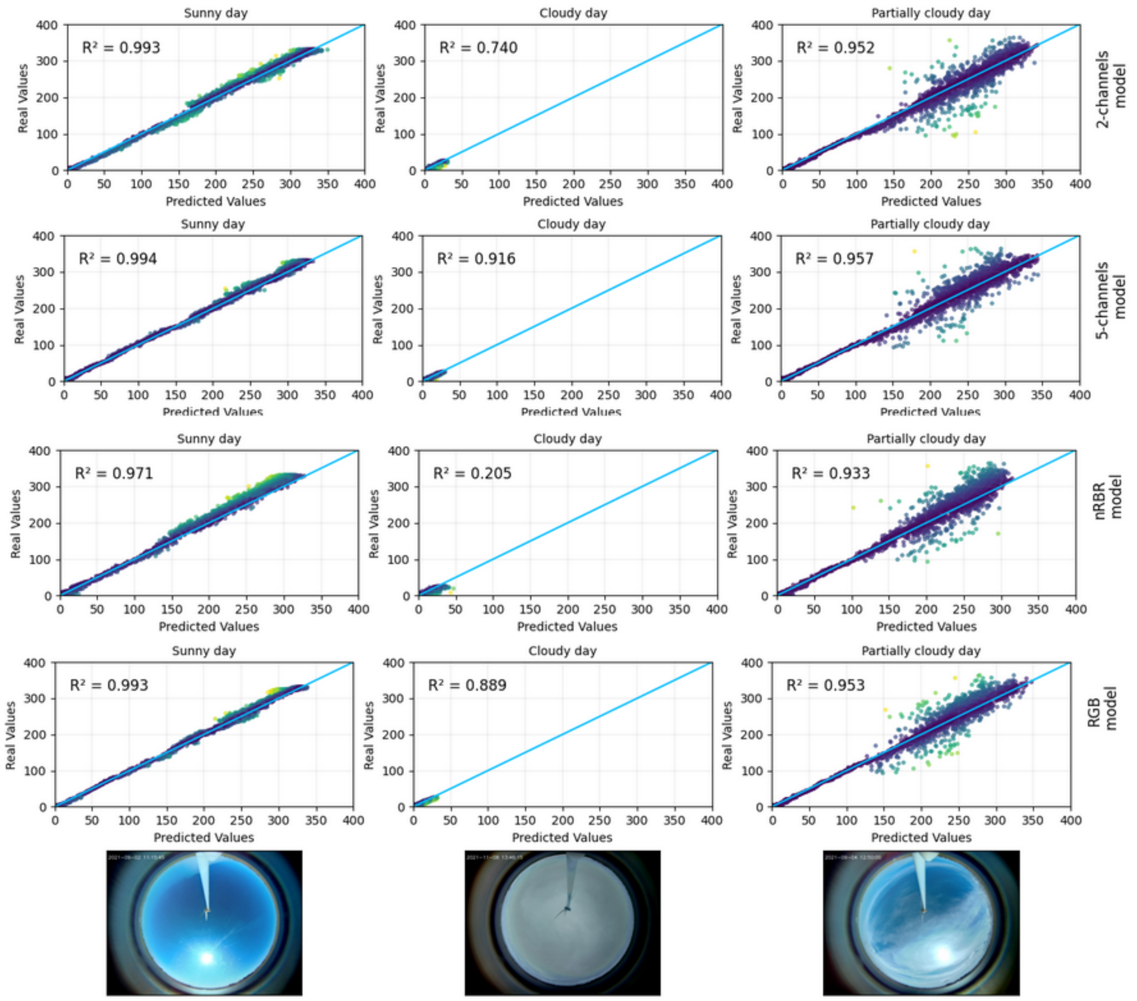


Figure 4.7: Linear regression between real and predicted data of each model. From left to right: sunny day, cloudy day and partially cloudy day. From top to bottom: 2-channels model, 5-channels model, nRBR model and RGB model.

This figure provides decisive information for the analysis of the results. As can be seen in the graphs corresponding to the nRBR, 2-channel and RGB models, the value of the coefficient of determination decreases notably on cloudy days, while in the 5-channel model this variation is not so noticeable and maintains the proportion of variation in the results that can be explained by the model above 90%. This result is of outmost importance, as it seems to show the effectiveness of the latter model in making sufficiently good predictions on sunny, partly cloudy and completely cloudy days.

Chapter 5

Conclusions

The prediction of the power produced by a photovoltaic plant through sky images is a very topical issue for the scientific community. An essential part of this process is the preprocessing of the images, which has been satisfactorily addressed in this work.

Four preprocessing techniques have been employed to treat the images that would be the input of a convolutional neural network. The results obtained in each of the cases have been evaluated with several error metrics that have allowed an informed comparison between them. The fact that in all situations very satisfactory metrics have been obtained, low values for MAE and RMSE and R^2 values close to 1, shows the efficiency of this type of neural networks in image analysis, and therefore, their potential in more complex models of photovoltaic prediction from sky images.

It is necessary to mention the rightward shift of the error distribution compared to a normal distribution. As mentioned in Sec.4.2, it implies that the models are making more prediction errors in a specific direction. This could be due to several factors, such as the lack of representation of certain sky types in the training data, the presence of noise or artefacts in the images, such as the windmill present in all images of the dataset, which affect the ability of the network to extract relevant features, or the lack of ability of the model to capture certain patterns in the data. In this case, it may be useful to perform a more detailed analysis of the training and validation data to better understand the causes of this bias and consider possible solutions, such as including more data, using techniques to remove obstacles in the images, or exploring different convolutional network architectures. This motivation is left for future work. This bias is not considered to have a significant impact on the final objectives of this work.

Based on the results of the comparisons between the applications studied, some interesting conclusions could be drawn. The 5-channels model has outperformed the traditional RGB model in terms of prediction quality. Taking into account its high computational cost compared to the other models, its use could be advantageous as long as sufficient resources are available.

The most competent models in this study are the RGB and the 5-channel models. As shown in Fig.4.5, the behaviour of both models is very similar, as they both obtained their maximum and minimum error peaks on the same day, and the rest of the peaks are very close in time. This event questions the usefulness of these two extra channels incorporated to the RGB images in the 5-channel model, because based on this result, this second model implies an inefficient increase in the computational cost, which in some cases may not be convenient. More precisely, according to the results obtained, the computational cost increases by 18.41% in the 5-channel model compared to the RGB model, while RMSE decreases by 5.16%, MAE by 13.14% and R^2 increases by a modest 0.10%.

The result that could determine the choice of the 5-channel model in future works is the one extracted from Fig.4.7. While the coefficient of determination in the RGB, 2-channel and nRBR models drops significantly for the cloudy day, this is not the case for the 5-channel model, which remains above 90%. Although it would be necessary to study what happens on other cloudy days and to particularise the results for different types of cloudiness, it could be concluded that these two extra channels added in the image preprocessing of the 5-channel model improve the detection of image details, being this improvement more remarkable on the type of days mentioned.

If this line of study is pursued further and the certainty of these results is confirmed, the benefit of this image preprocessing technique applied to more complex short-term photovoltaic prediction models, nowcasting, would be evident, since, as has been observed in various papers on this subject, [5, 8, 4], the great difficulty of these models lies in the prediction of photovoltaic power ramps, an event that occurs due to the cloudiness of the sky.

By focusing future studies on the potential of the 5-channel model, numerous improvements can be implemented. It would be interesting to compare the results of this 5-channel model with other models (with equally preprocessed images as input) whose architecture has been modified. It would also be possible to compare the results for different hyperparameter configurations in order to achieve a good fine tuning. Exploring with channels from other colour models may also be a possible avenue for improvement. Ultimately, the search for the ideal preprocessing of sky images for forecasting techniques is a very interesting topic that needs to be further studied.

References

[1] O. Morton, "A new day dawning?: Silicon Valley sunrise," *Nature*, vol. 443, pp. 19–22, Sept. 2006. Number: 7107 Publisher: Nature Publishing Group.

[2] IEA, "Technology roadmap solar photovoltaic energy - 2014 edition." https://iea.blob.core.windows.net/assets/3a99654f-ffff-469f-b83c-bf0386ed8537/pv_roadmap.pdf. Accessed: 2023-05-02.

[3] Our World in Data, "Solar PV cumulative capacity." https://ourworldindata.org/grapher/solar-pv-cumulative-capacity. Accessed: 2023-05-02.

[4] B. Juncklaus Martins, A. Cerentini, S. L. Mantelli, T. Z. Loureiro Chaves, N. Moreira Branco, A. von Wangenheim, R. Rüther, and J. Marian Arrais, "Systematic review of nowcasting approaches for solar energy production based upon ground-based cloud imaging," *Solar Energy Advances*, vol. 2, p. 100019, Jan. 2022.

[5] S. R. West, D. Rowe, S. Sayeef, and A. Berry, "Short-term irradiance forecasting using skycams: Motivation and development," *Solar Energy*, vol. 110, pp. 188–207, Dec. 2014.

[6] J. Merelo Guervós, "What is the difference between 'forecasting' and 'nowcasting' which can be used for linear assets or asset health monitoring and control?," 09 2016.

[7] "What is short term forecasting?." https://www.assembled.com/university/what-is-short-term-forecasting. Accessed: 2023-05-09.

[8] J. Zhang, R. Verschae, S. Nobuhara, and J.-F. Lalonde, "Deep photovoltaic nowcasting," *Solar Energy*, vol. 176, pp. 267–276, Dec. 2018.

[9] C. W. Chow, B. Urquhart, M. Lave, A. Dominguez, J. Kleissl, J. Shields, and B. Washom, "Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed," *Solar Energy*, vol. 85, pp. 2881–2893, Nov. 2011.

[10] J. Antonanzas, N. Osorio, R. Escobar, R. Urraca, F. M. de Pison, and F. Antonanzas-Torres, "Review of photovoltaic power forecasting," *Solar Energy*, vol. 136, pp. 78–111, 2016.

[11] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6, Aug. 2017.

[12] S. Saha, "A comprehensive guide to convolutional neural networks — the ELI5 way." https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53, Dec. 2018. Accessed: 2023-05-02.

[13] C. N. Long, J. M. Sabburg, J. Calbó, and D. Pagès, "Retrieving Cloud Characteristics from Ground-Based Daytime Color All-Sky Images," *Journal of Atmospheric and Oceanic Technology*, vol. 23, pp. 633–652, May 2006. Publisher: American Meteorological Society Section: Journal of Atmospheric and Oceanic Technology.

[14] Y. Chu, M. Li, C. F. Coimbra, D. Feng, and H. Wang, "Intra-hour irradiance forecasting techniques for solar power integration: A review," *iScience*, vol. 24, no. 10, p. 103136, 2021.

[15] M. Hasenbalg, P. Kuhn, S. Wilbert, B. Nouri, and A. Kazantzidis, "Benchmarking of six cloud segmentation algorithms for ground-based all-sky imagers," *Solar Energy*, vol. 201, pp. 596–614, 2020.

[16] T. Schmidt, J. Kalisch, E. Lorenz, and D. Heinemann, "Evaluating the spatio-temporal performance of sky-imager-based solar irradiance analysis and forecasts," *Atmospheric Chemistry and Physics*, vol. 16, pp. 3399–3412, Mar. 2016.

[17] GeeksforGeeks, "HSV color model in computer graphics." https://www.geeksforgeeks.org/hsv-color-model-in-computer-graphics/, August 2022. Accessed: 2023-05-02.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[19] Intel, "Redes neuronales convolucionales (CNN) y aprendizaje profundo." https://www.intel.es/content/www/es/es/internet-of-things/computer-vision/convolutional-neural-networks.html. Accessed: 2023-05-03.

[20] Orlando, "Sistemas de recomendación." https://www.grapheverywhere.com/sistemas-de-recomendacion-que-son-tipos-y-ejemplos/, Dec. 2019. Accessed: 2023-05-03.

[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

[22] Na, "Cómo funcionan las convolutional neural networks?- visión por ordenador." https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/, Nov. 2018. Accessed: 2023-05-05.

[23] M. Yani, S. Irawan, and C. Setianingsih, "Application of transfer learning using convolutional neural network method for early detection of terry's nail," *Journal of Physics: Conference Series*, vol. 1201, p. 012052, 05 2019.

[24] D. Scherer, A. Müller, and S. Behnke, "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition," in *Artificial Neural Networks – ICANN 2010* (K. Diamantaras, W. Duch, and L. S. Iliadis, eds.), (Berlin, Heidelberg), pp. 92–101, Springer Berlin Heidelberg, 2010.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[26] O. Boursalie, R. Samavi, and T. E. Doyle, *Evaluation Metrics for Deep Learning Imputation Models*. Cham: Springer International Publishing, 2022.

[27] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, M. Shahzad, W. Yang, R. Bamler, and X. X. Zhu, "A survey of uncertainty in deep neural networks," 2021.

Appendix A

Appendix

A.1 Results

Table A.1: Comparison of the performance metrics of the different models.

Model name	RGB	5-channels	2-channels	nRBR
Training loss [kW]	0.0244	0.0241	0.0271	0.0274
Training MSE [kW^2]	0.0017	0.0017	0.0021	0.0022
Validation loss [kW]	0.0145	0.0133	0.0161	0.0223
Validation MSE [kW^2]	0.0008	0.0007	0.0009	0.0013
Training time [min]	42.320	50.136	40.840	38.612
Validation time [sec]	1.972	2.311	1.819	1.701

Table A.2: Comparison of the quality metrics predictions between models.

Model name	RGB	5-channels	2-channels	nRBR
R^2 [adim]	0.990	0.991	0.988	0.980
MAE [kW]	5.831	5.154	6.371	9.194
RMSE [kW]	10.536	9.992	11.237	14.392

Table A.3: Comparison of the days of maximum and minimum prediction error for each model.

Model name	Max MAE day	Max MAE [kW]	Min MAE day	Min MAE [kW]
RGB	2021-09-04	13.90	2021-11-08	1.94
5-channels	2021-09-04	12.70	2021-11-08	1.71
2-channels	2021-09-25	14.24	2021-10-18	2.01
nRBR	2021-08-30	18.83	2021-09-30	3.65