electronics

MDPI

Article

Convolutional Neural Networks Used to Date Photographs

Jesús-Ángel Román-Gallego *, María-Luisa Pérez-Delgado and Sergio Vicente San Gregorio

Higher Polytechnic School of Zamora, University of Salamanca, Avenida de Requejo 33, 49022 Zamora, Spain; mlperez@usal.es (M.-L.P.-D.); sergiovsg99@usal.es (S.V.S.G.)
* Correspondence: zjarg@usal.es; Tel.: +34-980545000 (ext. 3745)

Abstract: Nowadays, the information provided by digital photographs is very complete and very relevant in different professional fields, such as scientific or forensic photography. Taking this into account, it is possible to determine the date when they were taken, as well as the type of device that they were taken with, and thus be able to locate the photograph in a specific context. This is not the case with analog photographs, which lack any information regarding the date they were taken. Extracting this information is a complicated task, so classifying each photograph according to the date it was taken is a laborious task for a human expert. Artificial intelligence techniques make it possible to determine the characteristics and classify the images automatically. Within the field of artificial intelligence, convolutional neural networks are one of the most widely used methods today. This article describes the application of convolutional neural networks to automatically classify photographs according to the year they were taken. To do this, only the photograph is used, without any additional information. The proposed method divides each photograph into several segments that are presented to the network so that it can estimate a year for each segment. Once all the segments of a photograph have been processed, a general year for the photograph is calculated from the values generated by the network for each of its segments. In this study, images taken between 1960 and 1999 were analyzed and classified using different architectures of a convolutional neural network. The computational results obtained indicate that 44% of the images were classified with an error of less than 5 years, 20.25% with a marginal error between 5 and 10 years, and 35.75% with a higher marginal error of more than 10 years. Due to the complexity of the problem, the results obtained are considered good since 64.25% of the photographs were classified with an error of less than 10 years. Another important result of the study carried out is that it was found that the color is a very important characteristic when classifying photographs by date. The results obtained show that the approach given in this study is an important starting point for this type of task and that it allows placing a photograph in a specific temporal context, thus facilitating the work of experts dedicated to scientific and forensic photography.

Keywords: photograph; artificial intelligence; convolutional neural networks; image recognition; classification

check for updates

Citation: Román-Gallego, J.-Á.; Pérez-Delgado, M.-L.; San Gregorio, S.V. Convolutional Neural Networks Used to Date Photographs. *Electronics* **2022**, *11*, 227. https://doi.org/10.3390/electronics11020227

Academic Editor: Gyu Myoung Lee

Received: 21 December 2021
Accepted: 8 January 2022
Published: 12 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In several areas, there are many photographs that are not dated, so it is very complicated to know when they were taken, regardless of the subject of the photograph (faces, people, cities, landscapes, buildings, etc.). Obtaining the exact date when they were taken is an arduous task. Currently, the use of digital photography is widespread. In these types of photographs, complementary information is stored in the EXIF (exchangeable image file) format data [1]. This information includes the date when the photograph was taken, its compression type, the file format and other characteristics that allow us to determine not only the date and place where a photograph was taken, but also the device used to take it. Therefore, in this case, it is easy to know the date when a photograph was taken. This is not the case with photographs printed on photographic paper, which, depending on the laboratory and the time when they were developed, may not contain the date of

printing. It is possible to calculate the approximate date a photograph was taken by making a comparison with other photographs, taking into account the landscape, the objects that appear in the photograph and even the clothes that people are wearing. The ability to date a photograph printed on photographic paper is important for various reasons. It can be interesting on a personal level, since it allows us to recognize people or moments from the past, but it is also very interesting professionally, especially in the fields of forensic photography, scientific photography and judicial photography [2]. Printed photographic images are of vital importance in this aspect because they are considered irrefutable evidence in certain cases.

Artificial intelligence (AI) applied to the field of image recognition is having a permanent and cutting-edge technological evolution, which allows using techniques of this type to optimize image processing and perform tasks that are very expensive or relatively impossible for people. This is the case with machine learning, which is commonly combined with computer vision. Computer vision includes methods to acquire, process and analyze all types of images in order to extract information for further automatic processing [3]. On the other hand, the goal of machine learning is to develop techniques that allow a computer system to learn autonomously [4]. Artificial neural networks (ANNs) are a specialized AI tool used to solve a wide variety of problems [5–12]. An ANN is a trainable system made up of a set of interconnected neurons organized in layers. To apply the network to solve a problem, it must be trained with a dataset that represents that problem. The training allows to adjust the weights of the existing connections between the neurons of the network. Once the training process is completed, the network will generate an output for each input presented to it.

Convolutional neural networks (CNNs) are a special type of ANN that have proven to be very effective in areas such as image recognition and classification [13–15]. CNNs take their name from one of their hidden layers: the convolutional layer. As noted above, these networks have become very popular in recent years due to the good results obtained in computer vision problems [16–22]. CNNs have a wide range of applications and are usually combined with other techniques to define systems that can solve interesting real-world problems. Recent applications include its use for the diagnosis of diseases [23–25], or its application in industry to automatically recognize manufacturing defects or the joining of parts and the optimization of these processes [26,27].

The architecture of CNNs is the same as that of multilayer ANNs, but in this case, the inputs are images. The hidden layers of a CNN are in charge of finding specific characteristics within the images that identify and differentiate them from the rest, which allows them to be grouped into classes.

Many authors have compared different well-known methods for image classification, such as support vector machines, k-nearest neighbor, fuzzy c-means or multi-layer perceptron [28–32], with CNNs being the ones that obtain the best overall results. It is also true that in certain cases, the results are similar to those obtained with other techniques. Taking into account the above, this study aims to design and implement a model based on CNNs capability of determining the year in which a photograph was taken. This operation is performed from the visual information contained in the image, without using the complementary data that would be available in a digital image. The study of CNNs as an image classification technique and its ability to classify photographs according to the year they were taken has practical applications in different fields including, but not limited to, scientific or forensic photography. Regardless of the difficulty of the problem, the developed model has obtained promising results that serve as a promising starting point from which to continue working to optimize its functionality when applied to this type of problem. An important contribution of our research is the application of an image recognition and classification technique to a problem that has been little dealt with in the literature. The chosen technique is CNN, which was shown to generate very good results in image processing. Although this technique has already been applied to image processing, the classification of the photographs is not performed in a conventional manner.

The proposed method performs an initial segmentation of each photograph. The segments are then processed by the network, which predicts a year for each segment. Finally, the year in which the photograph was taken is calculated as a weighted average of the year obtained for each segment.

The rest of the article is organized as follows. Section 2 presents a review of the work related to the proposal raised in the article. Section 3 introduces CNNs and their main characteristics. Then, Section 4 describes the data used for the experimentation and the process followed for their treatment by the neural network. Section 5 gives a detailed description of the tests and the modifications that were carried out on the convolutional models and the input photographs. Section 6 raises the points for discussion and, finally, Section 7 shows the conclusions obtained in this work.

2. Related Works

There are no current works that focus on the automatic classification of photographs according to the year in which they were taken.

However, different authors have published works on the automatic classification of historical photographs in which artificial intelligence is used to preprocess and classify photographs that allow archives and museums to organize these photographs [33]. The research described in [34] is applied to the automatic classification of landscape photographs based on the detection of vegetation and its changes over time. The article shows the possibility of applying a CNN for this task with good results. Mougiakakou et al. presented a study that focuses on the implementation of a computer system based on neural networks that allows the classification of landscapes according to their physiography, geological formations, vegetation and water resources, as well as by the various cultural interventions that occur in a given area [35]. The research presented in [36] describes algorithms capable of distinguishing photographs from graphic images, as well as classifying them into different types, such as real photographs, artistic photographs, images of scientific texts or comics. The studies presented in [37,38] focus on the classification of indoor and outdoor images based on well-known methods, such as the k-nearest neighbors algorithm and support vector machines, obtaining results that exceed 90% accuracy.

On the other hand, there are works that focus on automatic classification into semantic categories. Schettini et al. pointed out the difficulty of automatically classifying digital photographs into semantic categories, as well as the practical impact that optimizing this problem would have on the automatic detection of objects within the photographs, since there is an organization of the photographs into different semantic classes [39].

Other related works focus on different types of classification of photographs based on a series of criteria that involve the location of the photograph or the type of photograph. However, as noted above, there are not many studies that focus on a classification based on the year the photographs were taken, but there are several of them that date historical color images [40–42].

3. Convolutional Neural Networks Overview

To understand a little better how a CNN works, we compare it to human vision. For example, when a person sees a car, it is recognized because it has wheels, doors, headlights, etc. It could be said that by finding these components, a person claims to be seeing a car. The car may be missing a wheel, but still the person would say that it is a car because the rest of the characteristics still match those of a car and not something else. A CNN classifies objects in a similar way. Just as a human must first learn to identify a wheel or a door, a CNN does it through the learning algorithm. As might be expected, identifying the elements of a car is not enough, as car parts stacked in a pile contain all the elements but do not form a car. For this reason, a CNN must also learn how far elements are from each other, their relative sizes and their colors. Successive layers of a CNN find the features that identify the image. Each of the layers applies different levels of abstraction to detect different features. As the image progresses through the network, the features are

built hierarchically. In this way, there are features with low levels of abstraction and others with higher levels of abstraction. The initial layers of the network learn simple features, such as edges, while the subsequent layers learn more complex features, such as object shapes [14,43].

3.1. Convolutional Neural Network Layers

A CNN is a multilayer network that includes layers of different types. There are three basic layers that define a CNN: the convolution layer, the pooling layer and the softmax layer. They are specialized layers and are designed to perform a very specific task within the CNN. Figure 1 shows an example of a CNN architecture composed of two convolution layers, two pooling layers and a softmax layer. These layers are described in the following subsections.

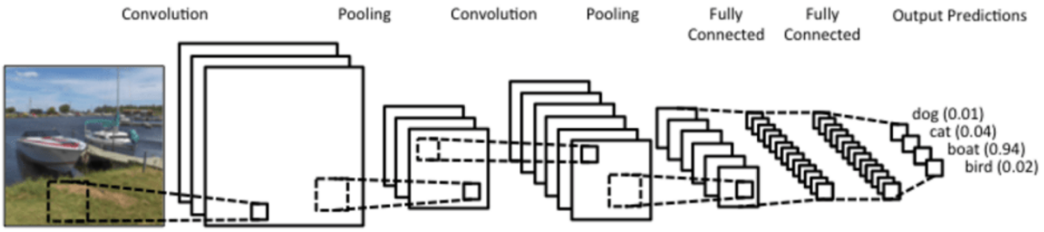

Figure 1. A basic convolutional neural network scheme.

3.1.1. Convolution Layer

The convolution layers of a CNN learn local patterns. This means that they learn a pattern at certain coordinates of the image and can find it elsewhere in the image.

A convolutional layer applies filters (also called kernels or feature detectors) to extract features. The output obtained as a result of the convolution of the image and a filter is called a feature map, convolved feature or convolved map. The convolution operation captures the local dependencies that exist in the original image.

The parameters that must be defined for this layer are the set of filters that can be learned and the size of the filters.

A filter has a specific size. It has a defined width and height, and extends to the full depth of the input volume. When working with images, the input volume is defined by the pixels in the image. When an RGB image is used, the depth is 3 (the number of colors used to represent each pixel in the image). The filter moves over the image from top left to bottom right. As this sliding is performed, the convolution operation is applied considering the content of the filter and the content of the image in the window on which the filter is positioned. Each convolution operation allows to calculate a value of the feature map. As the filter moves over the input volume, a two-dimensional activation map is produced that provides the responses of that filter at each spatial position.

In this way, the network learns filters that are activated when they detect some kind of visual feature represented by said filters. Such features are simple (edges, for example) when considering the initial layers of the CNN and become more complex (such as full circumferences) in deeper layers of the network. A filter that represents a specific feature lets us know if that feature appears in the image, how many times it appears and where it appears.

As an example, Figure 2 shows how a filter that detects vertical lines is applied to an image showing the letter "K", as well as the final result. To simplify the description, integer values are used. We consider an image of 9×9 pixels, which is represented by an integer matrix of 9 rows and 9 columns. In the original image (Figure 2a), the value -1 represents the background, and the value 1 represents a pixel of the figure corresponding to the letter. In this example, a 3×3 size filter is used (Figure 2b), which corresponds to the application of a mask to enhance the weight of the cells containing the searched figure. We denote $\overrightarrow{f}$ the vector that represents the filter: $\overrightarrow{f} = (-1, 4, -1, -1, 4, -1, -1, 4, -1)$. On the other

hand, we denote $\overrightarrow{v}$, the vector that represents the values of the 3×3 square marked in Figure 2a: $\overrightarrow{v} = (-1, -1, -1, -1, 1, -1, -1, 1, -1)$. The result of applying the convolution operation on the vector $\overrightarrow{v}$, which represents a specific area of the image, and the vector $\overrightarrow{f}$ that represents the filter, is $p = \overrightarrow{v} \cdot \overrightarrow{f} = 10$. The value calculated for p is one of those that define the feature map shown in Figure 2c (specifically, this value is highlighted in red).

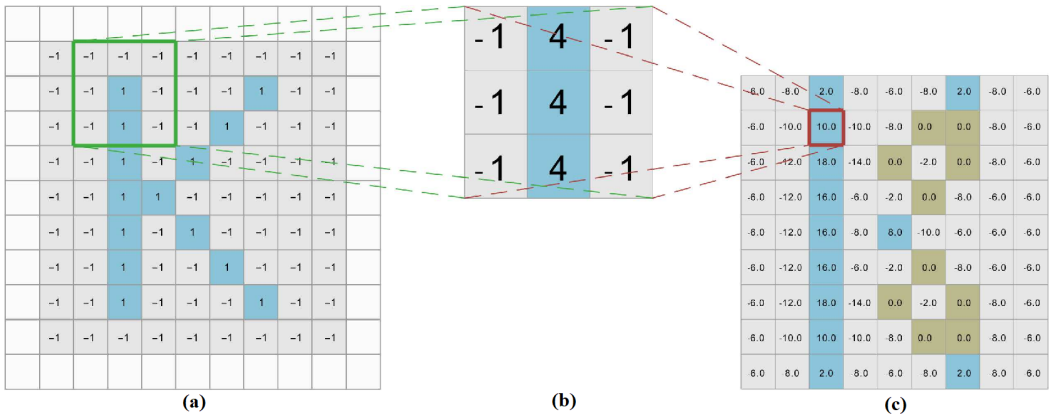

Figure 2. Detail of the convolution operation applied to an image representing the letter K. (**a**) Input image and the selected area of size 3×3; (**b**) Filter applied to the selected area; (**c**) Feature map obtained after applying the convolution operation to the input image. The result of the convolution applied to the window marked in green in the input image is highlighted in red.

We refer to *stride* as the number of rows and columns traversed per slide and the concept of *kernel* represents a matrix, which is slid over the image and multiplied by the input such that the output is enhanced in a determined, desirable way. When convolutions are applied to an image, the size of the result is reduced by an amount that depends on the size of the filter. To avoid this problem, a technique called *padding* is used, which consists in considering a margin around the input image. This operation allows the feature maps generated by the filter to have the same size as the original image.

If we consider the example in Figure 2, where the size of the input image is 9×9 and the size of the filter is 3×3, only 7 steps can be applied to slide the filter horizontally or vertically over the image, so the size of the resulting feature map will be 7×7. On the contrary, if a margin is included around the image whose size allows applying 9 steps, the result will be a feature map of size 9×9. The feature map shown in Figure 2c is calculated considering a margin of 1 pixel around the image of Figure 2a.

The convolution operation could consider a larger step to reduce the size of the image. It is also possible to include intermediate rows and columns in the image, as a padding, to make it larger.

Once a filter is passed over the whole image, a feature of the image is obtained. After applying successive filters, a set of feature maps is obtained. Figure 3 illustrates four possible filters for the previous example: horizontal, vertical and two diagonals. Figure 4 shows the feature map obtained by applying these filters to the image of the letter "K" shown in Figure 2a.

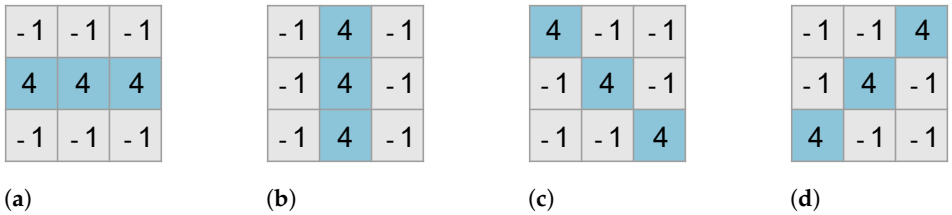

Figure 3. Filters. (**a**) Horizontal. (**b**) Vertical. (**c**) Diagonal 1. (**d**) Diagonal 2.

When looking at the feature map in Figure 4a, it is clear that the filter proposed in Figure 3a has not found any noticeable feature, since the largest values (highlighted with the blue background) are not large enough so that the signal that is sent to the next layer is significant. Therefore, this feature would not be valid and the filter would be modified by changing the weights in the backpropagation stage of the network. In contrast, the other 3 filters defined in Figure 3 show clear results: a vertical line (Figure 4b), a diagonal from northwest to southeast (Figure 4c) and a diagonal from southwest to northeast (Figure 4d).

Backpropagation is a well-known method used to update the weights of a neural network [44,45]. The backpropagation technique that a CNN uses differs from the basic technique in that it also uses convolution [46].

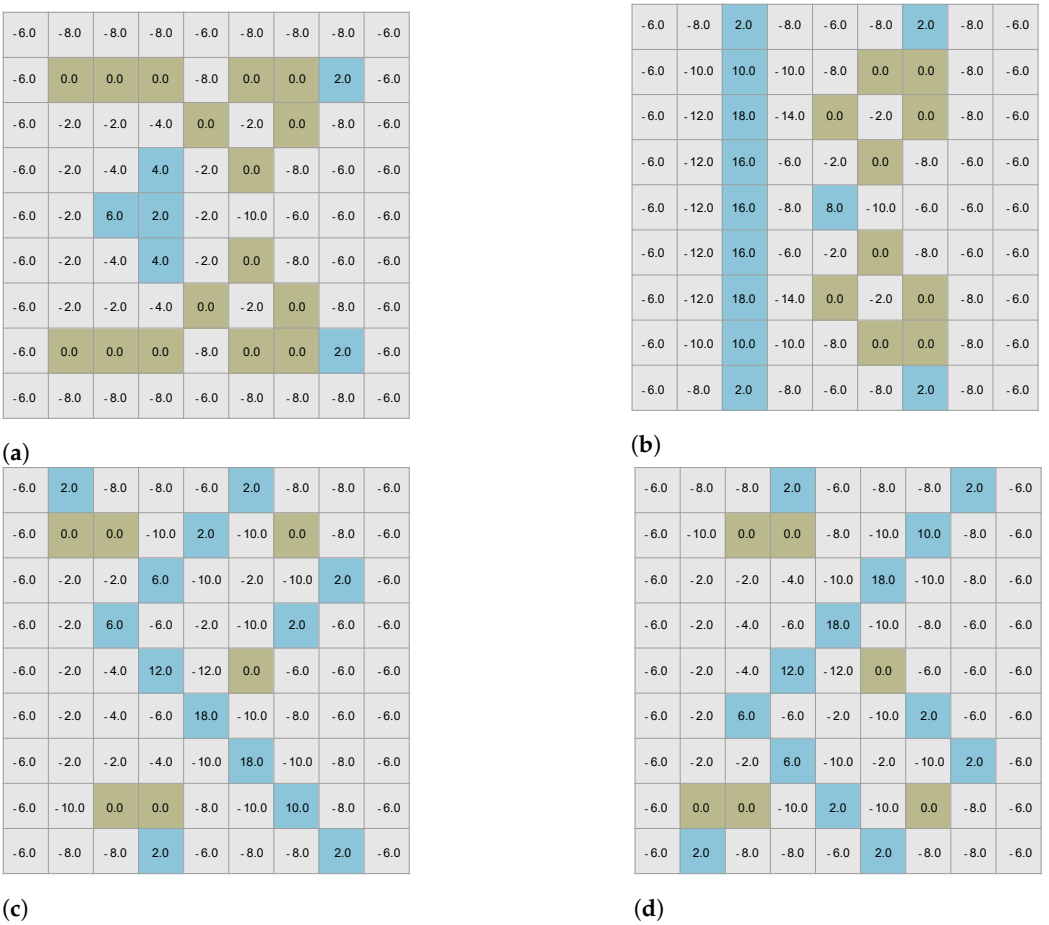

Figure 4. Convolution results for the filters included in Figure 3. (**a**) Horizontal. (**b**) Vertical. (**c**) Diagonal 1. (**d**) Diagonal 2.

To sum up, convolution layers apply filters to either the original image or a feature map of a CNN (depending on the position of the layer in the CNN architecture). A convolution layer applies several filters on its input to extract different features and learns those filters. Each filter generates different feature maps from the same original image.

3.1.2. Pooling Layer

The pooling layer is usually applied immediately after a convolution layer to reduce the size of the feature maps and condense the information for the following layers.

The operation performed in this layer is based on moving a window of a preset size across the length and width of the received feature, similar to how it is done in the convolution layer. The difference is in the way of operating on the affected elements and the non-overlapping of the window.

There are three basic types of operations that the window can perform to reduce the image size:

- min-pooling: this operation takes the minimum value found in the window.
- average-pooling: this operation computes the average of the values contained in the window.
- max-pooling: this operation takes the maximum value found in the window.

Figure 5 shows the result of applying the three pooling layers with a 3×3 window to the feature of the vertical filter sent from the previous layer. The border of each of the nine processed windows is shown in a different color, which is also used to frame the value resulting from the operation. Figure 5c,d shows the result of applying an average-pooling layer and a max-pooling layer, respectively, to the same example considered in Figure 5b.

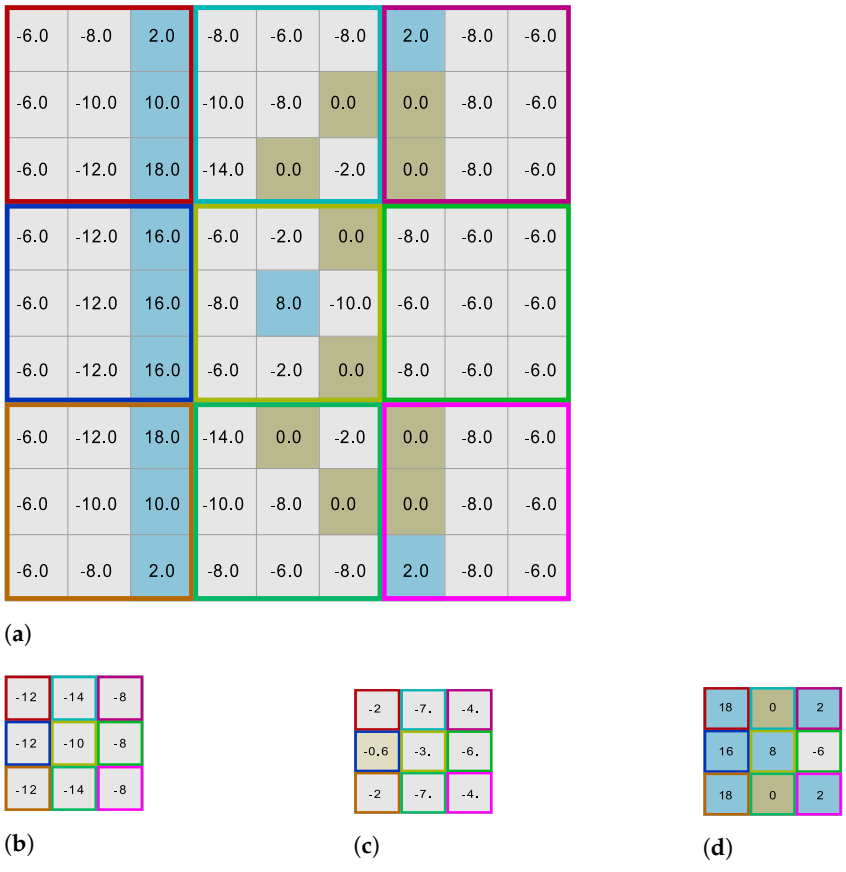

Figure 5. Pooling examples. (**a**) Feature map analyzed. (**b**) Min-pooling example. (**c**) Average-pooling example. (**d**) Max-pooling example.

3.1.3. Softmax Layer

The softmax layer is the last layer of a CNN and determines the class that the network associates with the image that was presented as input. It has as many outputs as classes of images that can be identified by the CNN. The layer determines the probability that an image belongs to each specific class.

This layer uses an input vector and an output vector of the same size. The elements of the input vector can take any real value, while the output vector includes real values whose sum is 1. Let $\overrightarrow{x} = (x_1, \ldots, n_n)$ denote the input vector of this layer, where n is the number of classes. Equation (1) defines the softmax function applied to the x_i element of the input vector. This function generates a value greater than 0, which is very small if the

input is negative and very large if the input is large. The softmax function highlights the differences among the input values, speeding up learning.

$$S(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad (1)$$

4. Dataset and Processing

Given the difficulty of the problem we are trying to solve, it is necessary to have a dataset large enough for the learning of the neural network to be meaningful. To this end, the dataset called "Date Estimation in the Wild Dataset" by Eric Müller, Matthias Springstein and Ralph Ewerth is used [47]. This dataset contains more than four hundred thousand photographs taken between 1960 and 1999.

The input images are represented using the *RGB* color model, which uses three channels to represent each pixel. These channels determine the amount of red (R), green (G) and blue (B) used to define each pixel in the image.

Once the dataset to be used in the study has been chosen, the three subsets that the CNN uses in the training, validation and testing stages must be defined. The training set is used to allow CNN to learn. The validation set is used to check if the network is learning correctly. The test set consists of data that was not used during network training. After concluding the training phase, the network generates an output for each data of the test set that is presented as input.

A subset of the images is selected to train and validate the neural network. Taking into account the number of photographs in the dataset and their characteristics, special care is taken in selecting the images to ensure that they are heterogeneous and contain specific elements, eliminating those that do not have a significant variety of tones and colors as predominant elements.

Since the photographs in the dataset are different sizes, it is necessary to apply preprocessing so that they are all the same size. This aspect is of utmost relevance for the training and validation process of the neural network. To make all the images presented as input to the CNN to have the same size, 300 × 300 pixel cutouts are extracted, as shown in Figure 6. The images whose size is smaller than the selected one are filled with a black background perfectly identifiable by the network. This operation homogenizes the input of the network.

Therefore, to define the set of images used for the training and validation phase, a 300 × 300 pixel frame is cropped in a random area of the image. In contrast, to define the subset for the test phase (Figure 7), the entire image is cut into pieces of the same size and all the pieces are presented to the network as independent images. This difference is established because it is assumed that it would be more accurate to estimate several images belonging to the same year and check if the network output is the same year for each fragment.

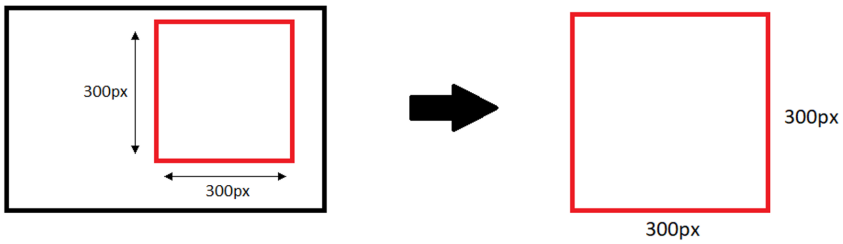

Figure 6. Image segmentation in training and validation phases.

The testing phase considers an initial set of 170 images of different sizes. Since each of these images is segmented into 300 × 300 pixel parts, the final set of images used in this phase includes more images (the total number of images depends on the size of the initial images).

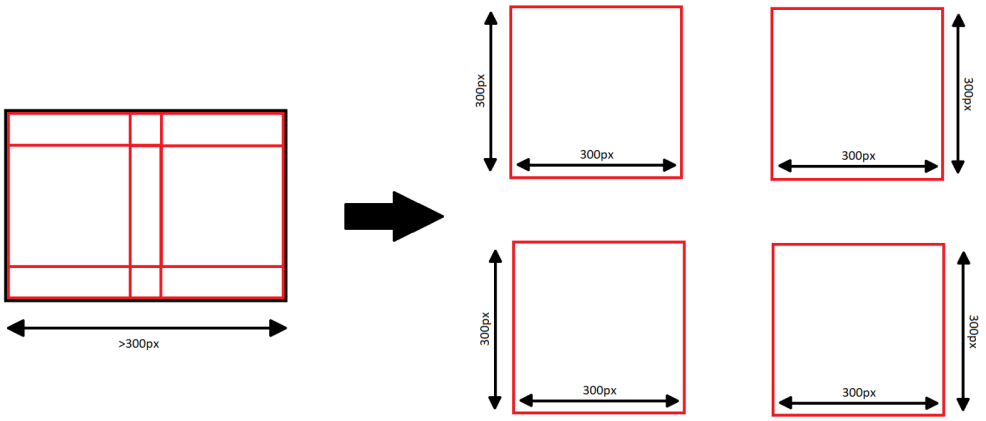

Figure 7. Image segmentation in test phase.

It is not recommended to test the network with images that are smaller than 300×300, as we would be "cheating" the network by passing it a "fake" image. This is because in this case, the image is filled with black until an image of the required size is obtained, as shown in Figure 8.

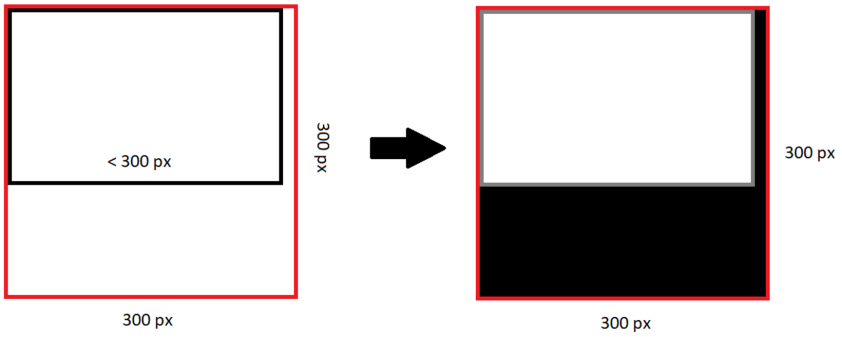

Figure 8. Segmentation of small images.

4.1. Network Output

The output of the CNN, and, therefore, the desired result, can be defined according to different methods:

- Linear output: The output is a 1×1 tensor with the year to which the analyzed image belongs.
- Binary output: The output is a vector whose size is equal to the number of years in the period in which the photographs of the dataset were taken. All the elements of the vector would have a value of 0, except the one corresponding to the year identified for the image being analyzed, which would have a value of 1.

Taking into account the characteristics of the problem we are analyzing, these two methods are not the most appropriate for this problem, since there is not a very noticeable difference between one year and another. Therefore, two other methods are evaluated, of which the second one is finally chosen.

- Binary output by blocks: The range of years of the dataset is divided into blocks of a certain size (five years, ten years, ...) and the output is a vector with one element for each block. The value 0 is assigned to all the elements of the vector, except the one corresponding to the block that contains the year that has been estimated for the photograph.
- Gaussian output: The output is a vector with one element for each possible year that indicates a range of certainty. Therefore, a tensor is used that stores values of a Gaussian function centered on the real year of the photograph, which approaches 0 when it moves away from the range that we want to accept as "reasonable".

Based on the previous discussion, the solution we propose has considered a Gaussian output. Therefore, to get the output of the network to be 1 for the desired year, it is sufficient to center the Gaussian on that particular year, as shown in Equation (2), where x is any year in which we want to evaluate the Gaussian function $g(x)$, xd is the reference year (the correct one, that is, the year the photograph was taken) and k is a constant that helps determine the width of the Gaussian function. Usually, k is assigned a negative value to get the Gaussian bell shape we want.

$$g(x) = e^{k(x-xd)^2} \tag{2}$$

If we denote D the distance with the reference year, $(x - xd)$, Equation (2) can be represented as shown in Equation (3).

$$g(x) = e^{k(D)^2} \tag{3}$$

If x coincides with the desired year, then D is zero, and the Gaussian function takes value 1 (Equation (4)), where k allows to control the opening of the Gaussian function and is calculated using Equation (5).

$$g(x) = e^{k(0)^2} = 1 \tag{4}$$

$$k = \frac{\ln(g(D))}{D^2} \tag{5}$$

On the other hand, the response interval of $g(D)$ must be fixed. In this case, we assume a variation of ± 5 years ($D = 5$), and we force the Gaussian function to return the value 0.05 at the extremes of this 5-year interval. Equation (6) shows the value computer for k.

$$k = \frac{ln(0.05)}{5^2} = -0.1198 \tag{6}$$

Figure 9 shows the graph obtained, where it can be observed that for $D = 5$, the value of the function is $g(D) = 0.05$.

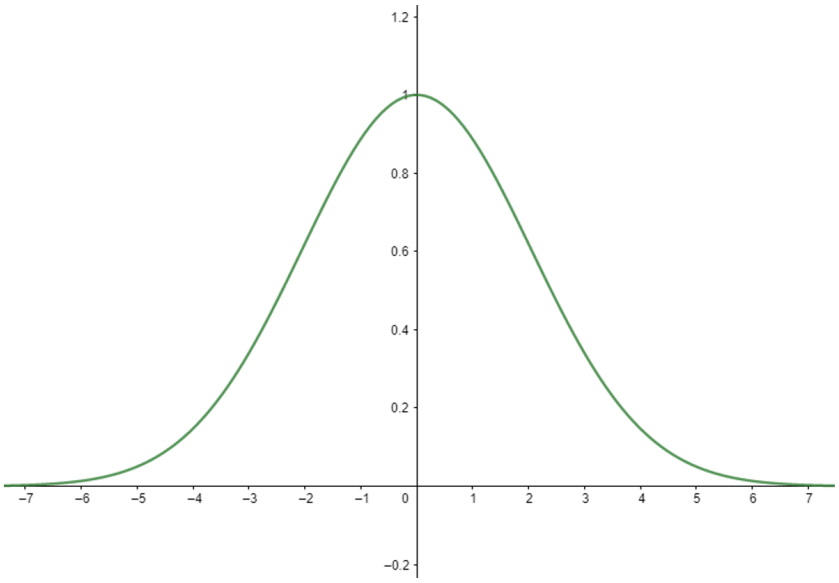

Figure 9. Representation of the Gaussian function with a difference of ± 5 years.

Once the photographs have been preprocessed, they are presented as input to the CNN and the network generates a year as output. Specifically, when a photograph is presented at the input, the year in which the photograph was taken is expected to be obtained. However,

since each image is segmented into several pieces, the network assigns a year to each of the segments into which the photograph is divided. The perfect result is obtained when the network interprets that the year in which all the pieces of an image were taken is the year in which the corresponding photograph was actually taken. Although it is practically impossible to obtain that perfect result, it is possible for close years to be obtained. For the year generated as output from the network, a score is also obtained that informs about the quality of the prediction made. The higher that score, the more likely it is that the photograph was taken in the year indicated by the network. In order to estimate an overall year for the whole photograph, we use a weighted average computed from the three best years of each segment and the associated scores. The best three years are those with the highest scores. The weighted average is computed by Equation (7), where $x1_i$, $x2_i$ and $x3_i$ are the three best years obtained for segment i, while $y1_i$, $y2_i$ and $y3_i$ are the scores that the network associates with these three best years. This equation generates the year that the network estimates for a complete photograph, denoted as α.

$$\alpha = \frac{\sum_{i=1}^{n} (x1_i y1_i + x2_i y2_i + x3_i y3_i)}{\sum_{i=1}^{n} (y1_i + y2_i + y3_i)} \quad (7)$$

4.2. Training, Validation and Testing Datasets

An appropriate dataset must be used so that the model can learn and generalize. For this reason, the original dataset is divided into three subsets, for each of the three phases of work with the network. Therefore, a set is defined for the training phase, another for the validation phase and another for the test phase. The training dataset consists of about 20 images per year for the years 1960 to 1999. The validation dataset consists of between 20% and 25% of the training dataset, that is, between four and five images per year. The test dataset is very similar to the training dataset.

The main reason why most of the data are used in the training stage is that the model has to learn to detect the essential features of the data. This allows the trained network to generalize what it has learned so that it can recognize the features of other new datasets. That is, the more data the network has for the training phase, the more different features it can learn that will later help the model to classify new unknown data.

5. Experimentation

To carry out the tests related to this research, it was decided to use the Python programming language with the OpenCV and PyTorch libraries [48–50] due to the versatility they offer when implementing this type of techniques. OpenCV (Open Source Computer Vision) is a library of open source computer vision functions [48]. It includes more than 2500 algorithms and is currently the largest computer vision library in terms of number of functions. This library allows processing images and recognizing objects within them, such as faces or figures. PyTorch is a framework for neural networks that allows the design and training of machine learning models, as well as their execution on the GPU (graphics processing unit) to accelerate computations [49,50].

The first operation that we must perform to solve a problem using a CNN is to define the architecture of the CNN. Next, the CNN must be trained with a dataset that first allows the network to learn and then allows the network to generalize those results to new datasets. An appropriate dataset must be used so that the model can learn and generalize. The previous section described the method applied to define the dataset used in each phase of CNN operation. Parameter selection and parameter optimization is an important objective and a challenge when implementing CNN-based techniques. Taking into account the objective of this work and assuming that there is no optimal method to identify the structure that increases the performance of these networks, in this work, the hyperparameters are defined as the structure of the network itself. For this reason, several models are proposed with modifications on the previous ones, based on the method described in [51] and its random search study.

All experiments carried out with the different proposed models and datasets were performed on a personal computer with AMD Ryzen 7 2700 8-core processor, GeForce gtx 1680 graphics card and 16 GB of RAM. The code for all the programs was compiled with Python 3.9. The CNNs training process is divided into epochs. An epoch is a complete training cycle in which each data in the training set is presented as input to the network and is processed by successive layers of the network until an output value is obtained. The number of epochs to use depends on the problem. In the tests carried out in our research, we initially considered 100 epochs and analyzed at which point the model could no longer improve the results. The applied experimental procedure consisted of testing several models, to select the one with the best results. To define new models, changes were made to the network architecture, always starting from the best solution found in the previous tests. That is, when in any of these modifications the solution obtained was worse or very similar to one obtained previously, the best solution previously found was taken again to define and test another model. All tested models were trained for 100 epochs.

For all models, the total training time was approximately 6 h, so there is no model that stands out in this regard in a significant way.

The tools used to design the CNN allow us to choose among various optimizers, such as RMSProp, AdaGrad, and ADAM [52]. We selected the last one, because it is the most used today. It is necessary to define the optimizer learning rate, which determines how much the layer values will change in each phase. In this case, we used the value 0.0001. A value too large for the learning rate may cause the network to bounce between high and low accuracy, while a value too small may cause training to take too long.

The image set is presented to the network to generate the output tensor and make predictions. To assess the quality of the prediction, the output is compared with the labels associated with the training images, which identify the year in which they were taken. The comparison is made using a loss function [53]. In our research, we selected the negative log-likelihood loss, commonly used in classification architectures. As mentioned above, the validation set consists of a small percentage of the available data. Hyperparameters are adjusted on this dataset, which are all parameters that are manually adjusted before the training process. Therefore, having a validation set allows us to find out which hyperparameters best fit our model. Finally, the test set has a percentage of data very similar to that of the validation set. This dataset is used to evaluate our model without fitting any parameters, using the parameters and hyperparameters fitted with the previous datasets to check to what extent the model is capable of giving adequate output for an input that had not been previously presented. To do this, we take as the input dataset the coordinates of the images to crop them into 300 × 300 pixel. As previously done, the images are batch-processed to generate a prediction, with the difference that now several predictions correspond to fragments of the same image, so the fragments of each image must be identified. The output returns the best three years for each image and a score to choose the years that the network identifies as correct. This information is used to calculate the weighted average of the best three years, and then the average of a single image is calculated from the weighted averages of all its fragments. The following sections show the results obtained with each architecture analyzed.

5.1. Initial Model

The architecture of the first model includes four convolutional layers and one linear layer. The first convolutional layer receives 3 channels of each RGB image and the convolution produces 128 output channels, with a *Kernel* of 5 and a *Stride* of 1. The second convolutional layer receives those 128 channels and generates 64 output channels, with a *Kernel* of 3 and a *Stride* of 2, so now the image is reduced to 150 × 150 pixels since it is halved. The third convolutional layer receives those 64 channels and produces 32 output channels, with a *Kernel* of 3 and a *Stride* of 2 (the image is now 75 × 75 pixels). The last convolutional layer receives the 32 channels from the previous layer and produces 8 output channels, with a *Kernel* of 3 and a *Stride* of 5, so now the image size is reduced to 15 × 15 pixels since

it is one-fifth of the previous size. The linear layer starts with the 8 output channels and the 15×15 size image that receives from the last convolutional layer and ends with 40 outputs, corresponding to the 40 years (between 1960 and 1999) in which the photographs that the CNN is going to classify were taken.

Figures 10 and 11 show the results obtained after completing the training of this architecture. The figures show the error obtained for the training and validation dataset, respectively, over the 100 training epochs.

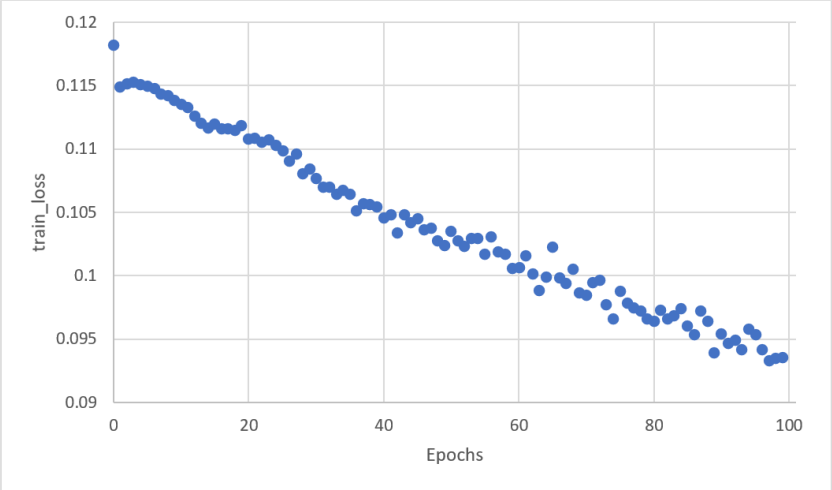

Figure 10. Representation of the training loss function for the initial model.

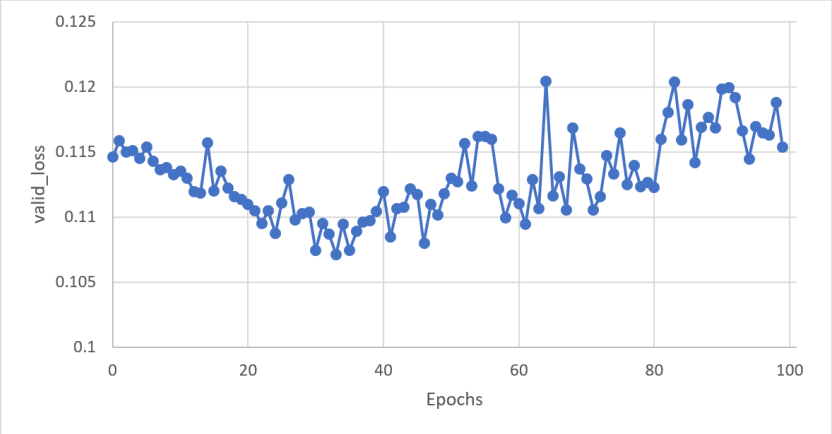

Figure 11. Representation of the validation loss function for the initial model.

As can be observed in Figures 10 and 11, the loss value for the training set decreases as the number of epochs increases. However, the value for the validation set reaches its best data (its lowest value) at epoch 33, and from that epoch, it follows an upward trend. Considering the 100 training epochs, the best loss value for the training set is 0.0933, while the best loss value for the validation set is 0.10708. Although they are close values, we can conclude that there is some overfitting. This means that the model fits the training data better than the validation data, that is, it does not generalize correctly because it is not able to classify all the unknown data.

Figure 12 shows a histogram that indicates the error (number of years) made by the network when estimating a year for the 669 images that were created with the proposed segmentation. It can be observed that the network correctly classifies 27 images (without any error), while for 52 images, the classification error is only 3 years. For 224 images, the classification error does not exceed 5 years. In addition, 52.46% of the images are classified with an error that does not exceed 10 years. Although there is a downward trend in the

number of images as the estimation error increases, there are some peaks in the histogram and the error is that there are too many years for some photographs.

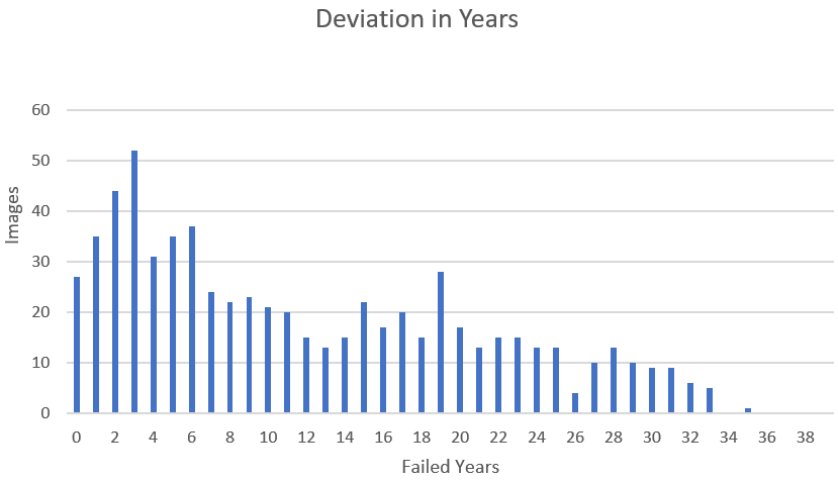

Figure 12. Histogram representing the failures of the initial model.

5.2. Second Model

The architecture of the second model includes the same convolutional layers as the first model, but includes one more linear layer. The first linear layer in this model starts with the same inputs as the linear layer in the first model and ends with 900 outputs. The second linear layer starts with the 900 inputs it receives from the hidden linear layer and ends with 40 outputs, corresponding to the 40-year period between 1960 and 1999.

Figures 13 and 14 show the training and validation errors corresponding to this model. As can be observed in the figures, the loss error of the training set follows a much faster downward trend than in the previous model. On the other hand, the loss error for the validation set reaches its best value in epoch 17, so it reaches this value much earlier than in the previous model. The best loss error for the training set over the 100 epochs is 0.0567, while the best value for the validation set is 0.10544. Therefore, a very high overfitting is observed in this model.

The histogram represented in Figure 15 shows that the network classifies 22 images without any error. On the other hand, the classification error does not exceed 5 years for 214 images. It should be noted that the first model correctly classified a slightly larger number of images than the second model, but Figure 15 shows that the second model has less dispersion with respect to more distant years.

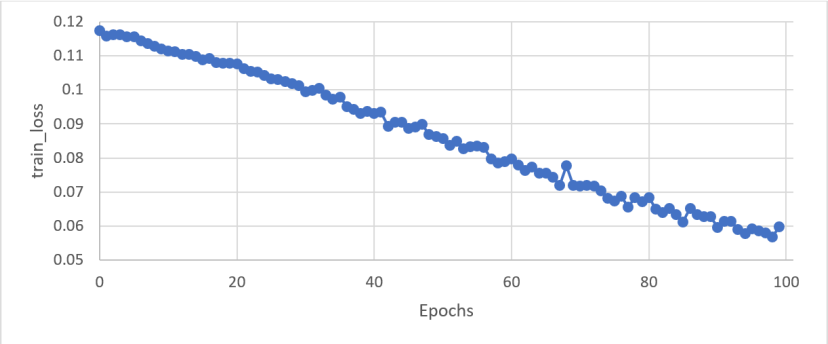

Figure 13. Representation of the training loss function for the second model.

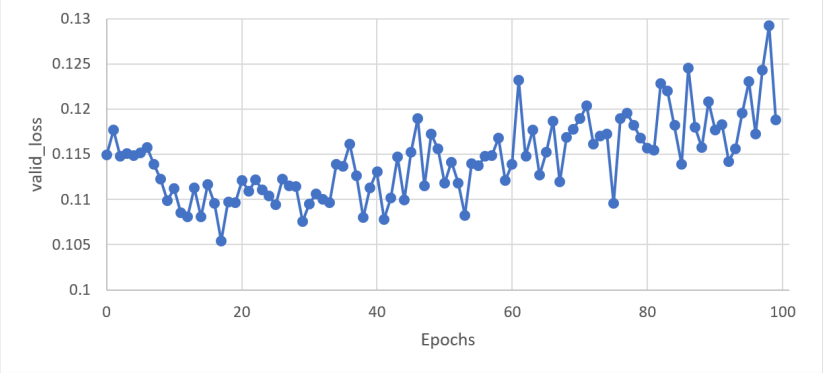

Figure 14. Representation of the validation loss function for the second model.

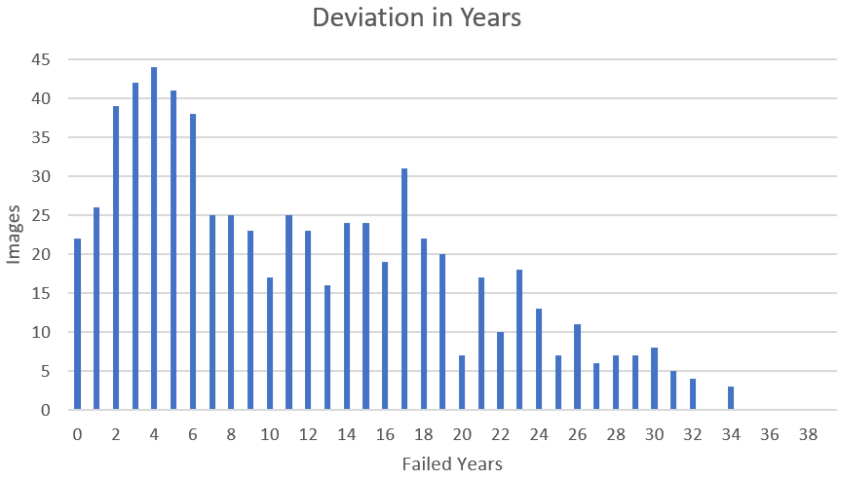

Figure 15. Histogram representing the failures of the second model.

5.3. Third Model

In order to try to improve the results of the previous model, the third model includes one more convolutional layer. This last convolutional layer receives 16 input channels and passes 8 output channels.

Figure 16 shows that the loss error for the training set follows a very similar trend to that of the previous model. However, Figure 17 shows that the loss error for the validation set is slightly larger.

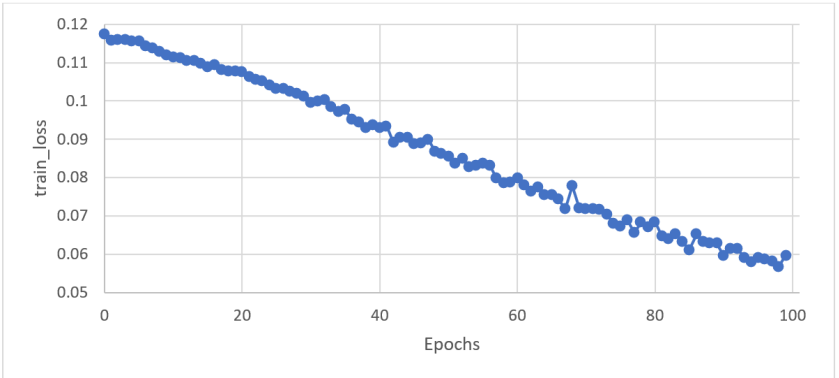

Figure 16. Representation of the training loss function for the third model.

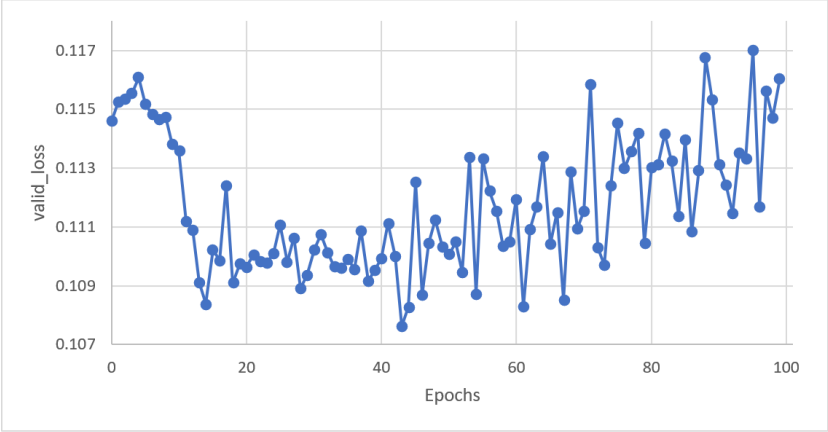

Figure 17. Representation of the validation loss function for the third model.

The histogram of this model (Figure 18) differs slightly from the previous model. Several quite sparse error peaks can be observed, as well as lower absolute accuracy than the previous model. This model classifies fewer images than the previous one with an error that does not exceed 5 years (205 photographs this model and 214 the previous model) but classifies more with an error that does not exceed 10 years (371 versus 342).

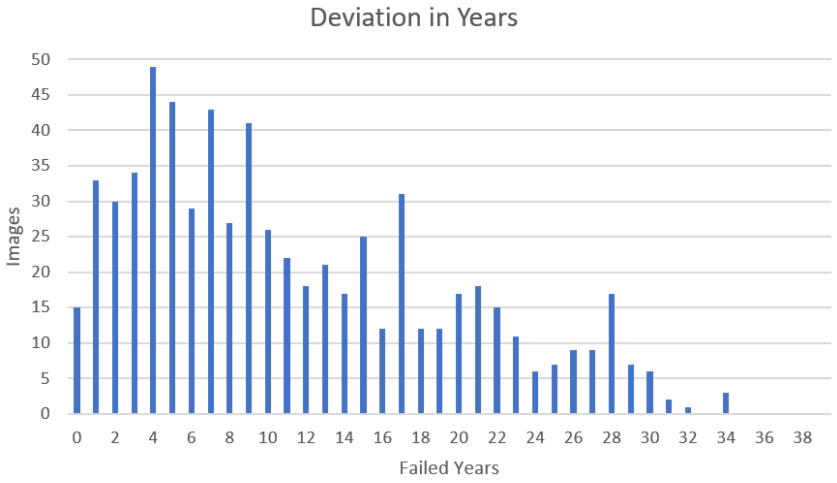

Figure 18. Histogram representing the failures of the third model.

One of the objectives of this work is to find a model that estimates the largest possible number of images with the least possible error. For this reason, the second model (described in Section 5.2) is taken as a reference to continue modifying the architecture in order to improve the results obtained.

5.4. Fourth Model

Starting with the second model, described in Section 5.2, we added one more linear layer that starts with the 450 outputs it receives from the last hidden linear layer and ends with 40 outputs.

Figures 19 and 20 show that the overfitting observed in model two tends to disappear in model four. Furthermore, the loss value obtained in the validation set is very similar in both models.

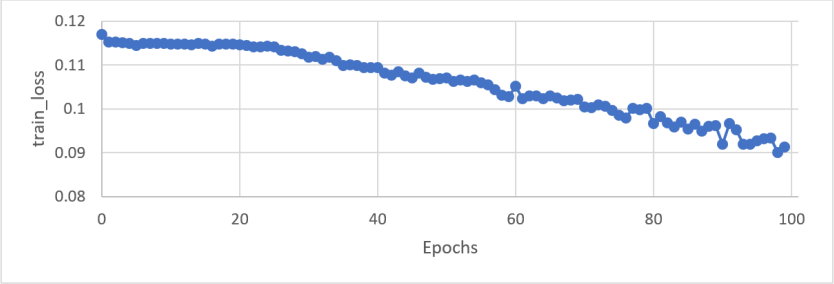

Figure 19. Representation of the training loss function for the fourth model.

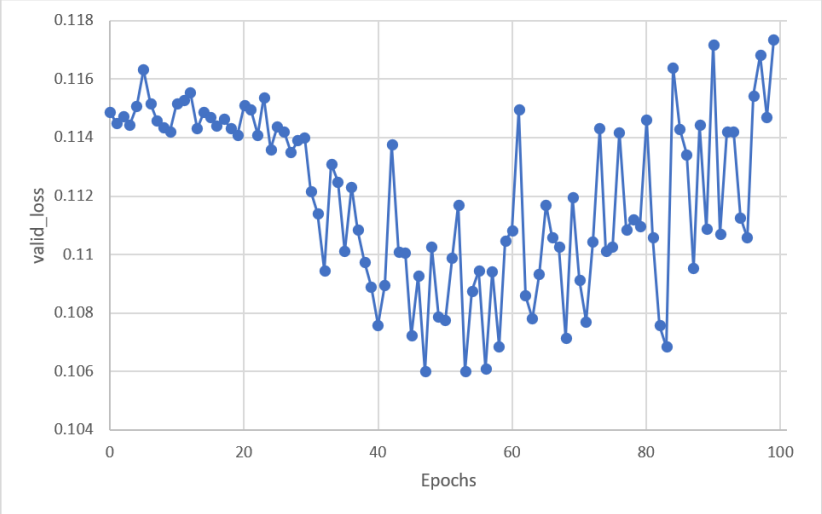

Figure 20. Representation of the validation loss function for the fourth model.

What at first might seem like a model that improves on the previous ones, does not obtain the expected results as shown in Figure 21. The figure shows a peak of accuracy with an error of 2 years, which is a good result. However, this model more frequently generates errors of more than 10 years. Certainly, this is the model that classifies the fewest photos with an error that does not exceed 10 years (49.03%) and the one that classifies the fewest photos without any errors (only 13).

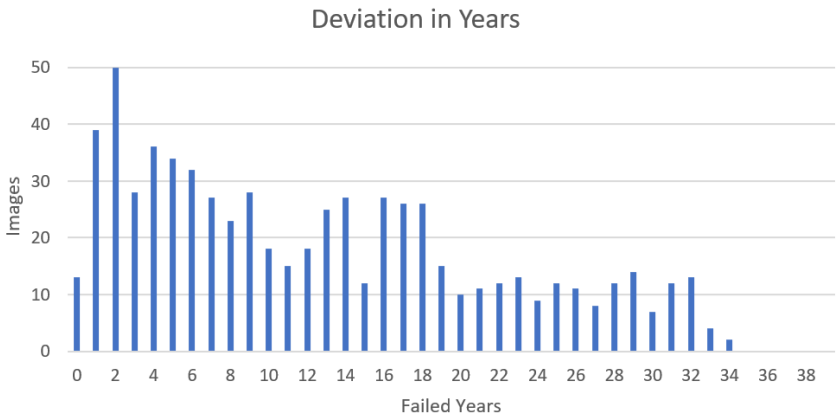

Figure 21. Histogram representing the failures of the fourth model.

In order to analyze the behavior of the proposed models, Table 1 summarizes the results obtained. It can be observed that there are three similar models and one very different. In view of the results, we discard the fourth model as the one with the worst results of those implemented. The third model classifies fewer images than its predecessors with an error of no more than 5 years, but classifies more images than those models with an

error of no more than 10 years. However, as mentioned above, the histogram of this model has a fairly large dispersion with respect to precision, so it is also discarded.

Table 1. Comparison of different models.

Model	Best	Best	Images with Classification Error		
	Training Loss	Validation Loss	$\leq$5 Years	(5, 10] Years	>10 Years
Initial Model	0.09330903756	0.1070820068684	224	127	318
Second Model	0.05672707886	0.1054419262115	214	128	327
Third Model	0.08476000656	0.1070820068684	205	166	298
Fourth Model	0.09006916169	0.1059944923857	200	128	341

There are no notable differences in the results obtained with the first two models. However, we selected the second model as the most appropriate because it obtained lower loss values and the histogram shows less dispersion of errors. Since the selected models (initial and second models) give similar results, a new test was carried out to obtain a final model. For this purpose, both models were retrained for 100 epochs, but larger training and validation sets were used. Specifically, the training set used includes 4000 images (100 per year) and the validation set includes 933 images.

Figures 22 and 23 show the histograms corresponding to each of the models selected as finalists. After training the two models for 100 epochs, good results are obtained in both cases, as shown in Table 2. The peak of the first model corresponds to an error of four years, while in the second model, it corresponds to an error of three years. However, the first model classifies 411 photographs with an error that does not exceed 5 years, while the second only achieves it for 369 photographs. This means that the first model shows more greater asymmetry and, therefore, a better distribution, classifying fewer images with an error that we consider large (>10 years).

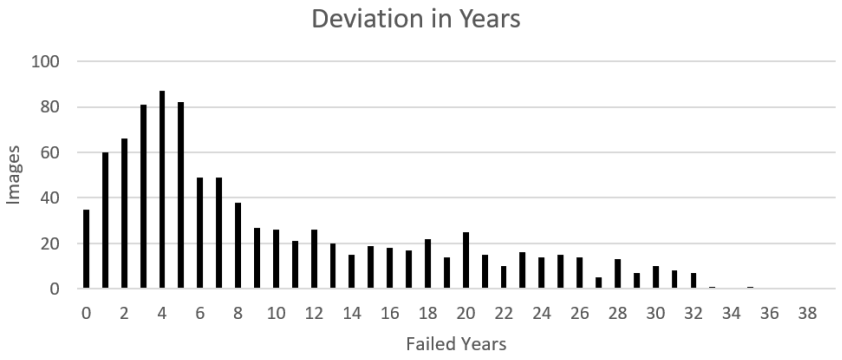

Figure 22. Histogram representing the failures of the first model (for the extended dataset).

Figure 23. Histogram representing the failures of the second model (for the extended dataset).

Table 2. Comparison between initial model and second model with 100 epochs of training (for the extended dataset).

Model	Best	Best	Images with Classification Error		
	Training Loss	Validation Loss	$\leq$5 Years	(5, 10] Years	>10 Years
Initial Model	0.09867622717	0.1058833766225	411	189	333
Second Model	0.09532483934	0.1089749876550	369	225	339

In view of the results obtained with the previous experimentation, the first model was chosen as the basis for this work. Subsequently, an analysis was carried out to determine the images that generate more classification errors and the cause of such errors. It should be noted that the 933 images processed by the network are not complete images, but pieces of 300 $\times$ 300 pixels in which the photographs were previously segmented. Therefore, we actually only used 237 images for the network to interpret as independent images. For this reason, we analyzed if all the pieces of the same image were classified in the same year or if they were classified in different years.

5.5. Analysis of Photographs Grouped by Decades

This section analyzes the images grouped by decades, in order to have a perspective of what types of photographs are more difficult to classify. Figure 24 shows the histogram of the images corresponding to the 1960s. As can be observed, the network correctly classifies most of the images improving the estimation considerably after 5 years for which, as we have considered, an error of $\pm$5 years is considered a good prediction.

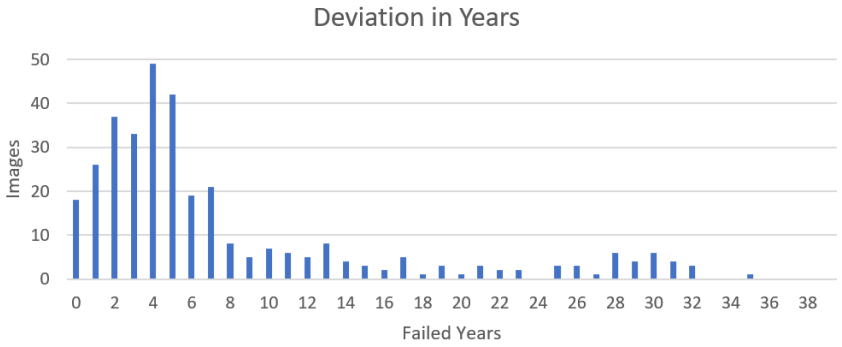

Figure 24. Histogram representing the failures for images from the 1960s.

Figures 25 and 26 are two examples of photographs for which the model has failed most significantly. All the images for which it has failed in more than 10 years share color as the main characteristic. It should be noted that most of the photographs from this decade used to train the network are black and white photographs. This may be the reason why the network fails with the images in Figures 25 and 26.

Figure 25. Photographs corresponding to the 1960s (1).

Figure 26. Photographs corresponding to the 1960s (2).

Figure 27 depicts the graph of classification failures in the 1970s. It is very similar to the previous one, with no appreciable improvement. It can be observed that the peak error is three years and that few images are classified with an error greater than 10 years. In this case, the network has classified most of the photographs with less than 10 years of error, except for photographs of the type shown in Figure 28.

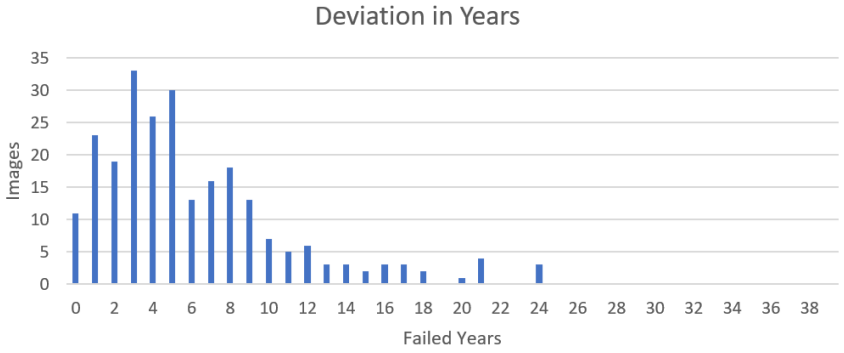

Figure 27. Histogram representing the failures for images from the 1970s.

For the first photograph in Figure 28 the network classifies all the pieces around the year 90. Taking into account these results, it can be thought that the network gives a lot of weight to the color of the photographs to perform the classification.

Figure 28. Photographs corresponding to the 1970s.

From the 1980s onwards, the results vary significantly. The histogram shown in Figure 29 includes several peaks, reaching the maximum value when the error is 6 or 20 years. Figures 30 and 31 show misclassified photographs from the 1980s.

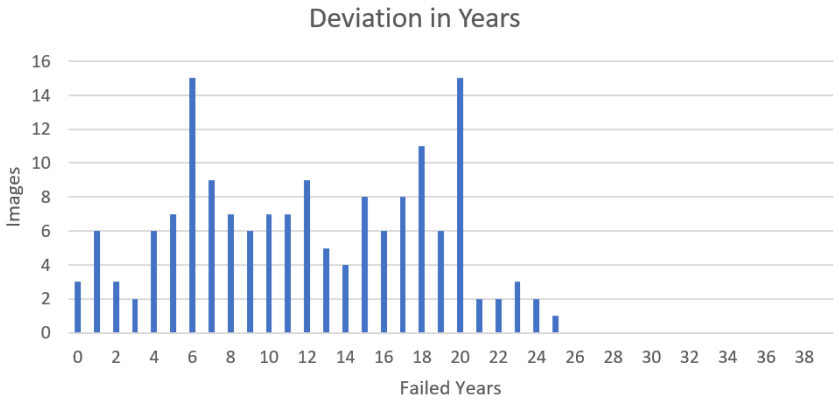

Figure 29. Histogram representing the failures for images from the 1980s.

Figure 30 shows a dark image, with many completely black parts. For this reason, the network does not completely differentiate these types of photographs, since it considers that the images with black and white tones correspond to the 1960s. Although it is true that the central part of the image is classified correctly, the weighted average for the entire image gives a date with a considerable error. For the photographs represented in Figure 31, the model classifies all its pieces in the 1960s because they are black and white photographs or include tonalities without significant color contrasts.

Figure 30. Photograph corresponding to the 1980s (1).

Figure 32 shows the histogram corresponding to the 1990s. The photographs shown in Figures 33 and 34 are classified with a large error because the model places them between the years 1968 and 1972, respectively; that is, the error is approximately 20 years. If we look closely, we can see that they are very similar images, with a great predominance of green and grayish colors. These tonalities are typical of the photographs of that time. However, an important detail in the image of Figure 34 is that the model classifies the pieces it analyzes in different years. As explained above, this may be due to the homogeneity in the colors of some areas.

(**a**) (**b**)

Figure 31. Photographs corresponding to the 1980s (**a**) and (**b**).

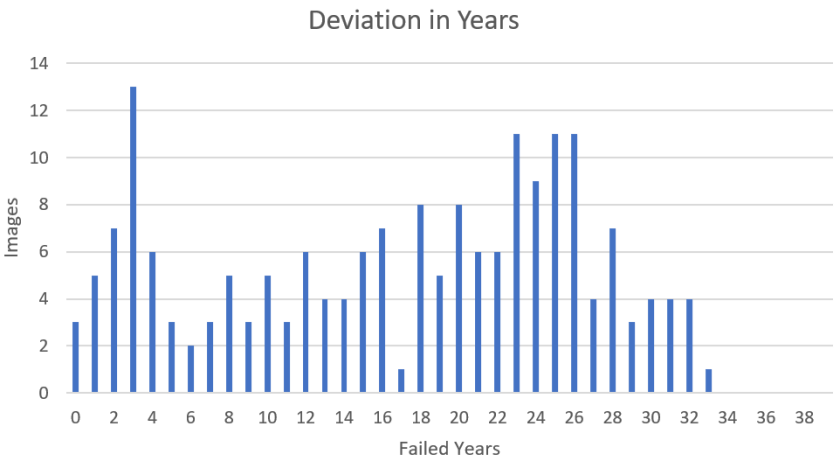

Figure 32. Histogram representing the failures for images from the 1990s.

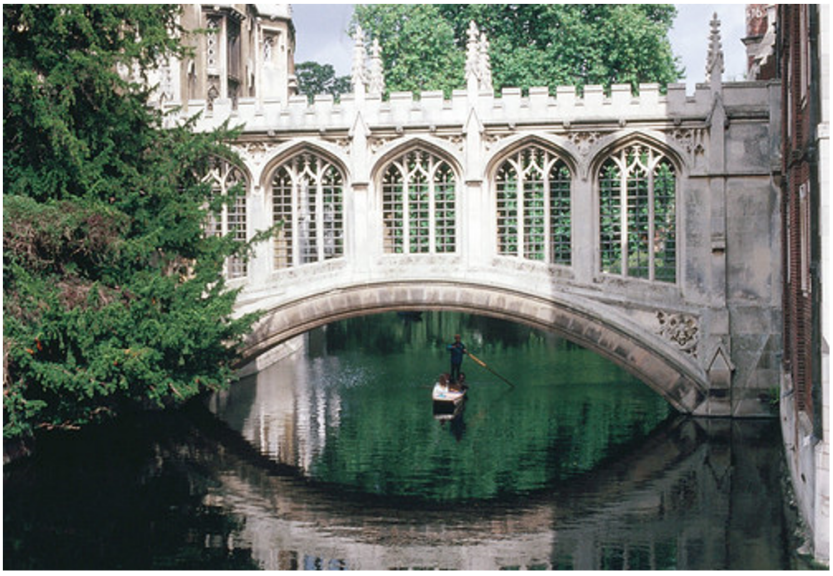

Figure 33. Photograph corresponding to the 1990s (1).

Figure 34. Photograph corresponding to the 1990s (2).

5.6. Model Evaluation with Respect to Color

After analyzing the results obtained in each of the decades, as well as the photographs for which more error is produced at the time of estimation, it can be determined that color is a characteristic with great weight in determining the classification. For this reason, it was decided to evaluate the model using the same image in color and grayscale to be able to conclude if color is a determining characteristic in this work. Figure 35 shows a photograph corresponding to the year 1981, including a color version and a black and white version.

(**a**) (**b**)

Figure 35. Photographs corresponding to 1985, in color (**a**) and grayscale (**b**).

When introducing the photographs of Figure 35 in the proposed model, it estimates that the color photograph is from 1983, while for the black and white photograph, the model estimates that it is from 1965. Taking into account the results obtained with the color variation, an analysis of the influence of the image quality is carried out. For this purpose, the model is tested with an image from the 1980s to subsequently lower the quality and observe the behavior of the network.

The photographs shown in Figure 36 correspond to the year 1983. The model classifies subfigure (a) in 1982. Subfigure (b) was generated by adding a noise filter and a dust filter. It is observed that the network classifies subfigure (b) in 1980, so the estimation error is not significant. However, although quality is not as important as color, this feature must be taken into account in our problem. Finally, a color modification is proposed in the same photograph to test the behavior of the model. Specifically, three different shades are considered: greenish, light and dark (as shown in Figure 37).

(**a**) (**b**)

Figure 36. Photographs corresponding to 1983, original (**a**) and low quality (**b**).

(**a**) (**b**) (**c**)

Figure 37. Photographs corresponding to 1983 with different. (**a**) Greenish; (**b**) light; (**c**) dark.

When the images shown in Figure 37 are entered into the model, it classifies image (a) in 1975, (b) in 1977, and (c) in 1968. The tests carried out with the model allow us to conclude that color is a determining characteristic in our work, so it is necessary to take it into account when classifying photographs.

6. Discussion

The research described in this article proposes the application of CNNs to the classification of photographs by the year in which they were taken. This work defines a starting point for further research and application of the established basis for the dating of analog images to facilitate the work of professionals in the field of scientific and forensic photography, among others. The dataset used to carry out this work includes photographs taken between 1960 and 1999, corresponding to 40 different classes (one class for each year). Between 75% and 80% of the images were used to define the training dataset, and between 20% and 25% were used to define the validation dataset. Four different models were trained by modifying their hyperparameters in order to obtain an optimal result. These models were evaluated and their results are summarized in Table 1. The analysis of the results indicates that models 1 and 2 are the ones with the best results. Taking these models as a reference, they were trained with a larger dataset. As shown in Table 2, model 1 is the one that presents the best classification results for the extended dataset. Based on this result, model 1 was taken as a reference and a new analysis was performed for each decade. It was observed that in the older decades, the classification was better; as the photographs are obtained with greater sharpness and a greater number of colors, the CNN assigns the photographs in years closer to the 1980s and 1990s. Due to these results, an analysis and evaluation of the model based on the color of the photographs was proposed, in which noise and distortion were introduced. From this last analysis, it was concluded that color is a definitive determining characteristic when obtaining the year in which these photographs were taken.

Taking into account that there are no works focused on solving the problem considered in this article, the priority of our research was to find a model capable of classifying the photographs according to the year in which they were taken, regardless of performance in terms of processing time. Certainly, this should be taken into account in future works so that different classification techniques can be used.

In addition, it should be noted that obtaining an exact classification in this problem is very difficult, due to the quality of the camera used to take the photograph. The cameras last for several years, and the photographs taken within a few years (for example, in an interval of 5 years) do not have important differences. In addition, professional cameras are very different from non-professional ones, so an older photograph taken with a professional camera would seem more modern than a newer photograph.

Finally, the oldest photographs vary greatly depending on their country of origin. The degree of industrialization and economic development of the countries was more unequal at that time and this conditioned the type of devices available in each case. For example, in the 1960s and early 1970s, in Spain, most of the photographs were in black and white, while in other countries, such as the United States, the United Kingdom or Japan, they were already in color.

7. Conclusions

The main objective of this work was the analysis of convolutional models applied to the recognition of photographs in order to estimate the year in which they were taken. When working with the photographs, we performed a chunked segmentation from a fixed-size window for training, but we used the entire image in the test phase, in order to achieve greater generalizability of the model. The variation of the proposed network architecture implies an important modification in the results obtained by the final model. However, in many cases, this factor is not as decisive as the dataset used, the typology of the photographs and their characteristics. In view of the experiments carried out, obtaining a model with a practically zero error rate is very complicated. This is partly due to the fact that the devices with which the images are taken have a useful life of several years, and the photographs in a range of ± 5 years do not show significant differences. In addition, the quality of professional devices is very different from that of non-professional devices, so an old photograph taken with a professional camera has much higher quality and, therefore, it can be classified in years after the corresponding one. Photographs from several decades ago vary greatly depending on their country of origin, as globalization was at a very early stage. In the 1960s and early 1970s, most photographs in many countries were in black and white, while in other more advanced countries (such as the United States, the United Kingdom or Japan) they were already in color. The analysis of the experimental results allows to conclude that the network gives great importance to the color of the photographs over other characteristics. Therefore, the results of the model are very accurate in years where most photographs are in black and white or include very characteristic dark colors. On the contrary, the largest errors are obtained for color photographs, so that the model has a lower success rate in the 1980s and 1990s. The model developed has obtained promising results, so we will continue working on it to optimize its functionality.

Author Contributions: Conceptualization, J.-Á.R.-G. and M.-L.P.-D.; methodology, J.-Á.R.-G., M.-L.P.-D. and S.V.S.G.; software, S.V.S.G.; validation, J.-Á.R.-G., M.-L.P.-D. and S.V.S.G.; formal analysis, J.-Á.R.-G., M.-L.P.-D. and S.V.S.G.; investigation, J.-Á.R.-G., M.-L.P.-D. and S.V.S.G.; resources, J.-Á.R.-G., M.-L.P.-D. and S.V.S.G.; data curation, J.-Á.R.-G., M.-L.P.-D. and S.V.S.G.; writing—original draft preparation, J.-Á.R.-G. and M.-L.P.-D.; writing—review and editing, J.-Á.R.-G. and M.-L.P.-D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was carried out in collaboration with the HP Technology Observatory.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Camera & Imaging Products Association; JEITA. *Exchangeable Image File Format for Digital Still Cameras: EXIF Version 2.32*; JEITA: Tokyo, Japan, 2019.
2. Rohatgi, R.; Kapoor, A. Importance of still photography at scene of crime: A forensic vs. judicial perspective. *J. Harmon. Res. Appl. Sci.* **2014**, *2*, 271–274.
3. Gabel, V.P. *Artificial Vision*; Springer: Munich, Germany, 2017.
4. Molnar, C. *Interpretable Machine Learning*; Lulu. com: Raleigh, NC, USA, 2020.
5. Abduljabbar, R.; Dia, H.; Liyanage, S.; Bagloee, S.A. Applications of artificial intelligence in transport: An overview. *Sustainability* **2019**, *11*, 189. [CrossRef]
6. Mellit, A.; Kalogirou, S.A. Artificial intelligence techniques for photovoltaic applications: A review. *Prog. Energy Combust. Sci.* **2008**, *34*, 574–632. [CrossRef]
7. Ullah, Z.; Al-Turjman, F.; Mostarda, L.; Gagliardi, R. Applications of artificial intelligence and machine learning in smart cities. *Comput. Commun.* **2020**, *154*, 313–323. [CrossRef]
8. Román, J.Á.; Pérez-Delgado, M.L. A Proposal for the organisational measure in intelligent systems. *Appl. Sci.* **2020**, *10*, 1806. [CrossRef]
9. Hosny, A.; Parmar, C.; Quackenbush, J.; Schwartz, L.H.; Aerts, H.J. Artificial intelligence in radiology. *Nat. Rev. Cancer* **2018**, *18*, 500–510. [CrossRef]
10. Zhang, D.; Han, S.; Zhao, J.; Zhang, Z.; Qu, C.; Ke, Y.; Chen, X. Image based forest fire detection using dynamic characteristics with artificial neural networks. In Proceedings of the 2009 International Joint Conference on Artificial Intelligence, Hainan, China, 25–26 April 2009; pp. 290–293.
11. Basheer, I.A.; Hajmeer, M. Artificial neural networks: Fundamentals, computing, design, and application. *J. Microbiol. Methods* **2000**, *43*, 3–31. [CrossRef]
12. Capizzi, G.; Lo Sciuto, G.; Napoli, C.; Shikler, R.; Woźniak, M. Optimizing the organic solar cell manufacturing process by means of AFM measurements and neural networks. *Energies* **2018**, *11*, 1221. [CrossRef]
13. Khan, S.; Rahmani, H.; Shah, S.A.A.; Bennamoun, M. A guide to convolutional neural networks for computer vision. *Synth. Lect. Comput. Vis.* **2018**, *8*, 1–207. [CrossRef]
14. Albawi, S.; Mohammed, T.A.; Al-Zawi, S. Understanding of a convolutional neural network. In Proceedings of the 2017 International Conference on Engineering and Technology (ICET), Antalya, Turkey, 21–23 August 2017; pp. 1–6.
15. O'Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458.
16. Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y. Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. *Sensors* **2017**, *17*, 818. [CrossRef]
17. Gopalakrishnan, K.; Khaitan, S.K.; Choudhary, A.; Agrawal, A. Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Constr. Build. Mater.* **2017**, *157*, 322–330. [CrossRef]
18. Luo, H.; Xiong, C.; Fang, W.; Love, P.E.; Zhang, B.; Ouyang, X. Convolutional neural networks: Computer vision-based workforce activity assessment in construction. *Autom. Constr.* **2018**, *94*, 282–289. [CrossRef]
19. Hongtao, L.; Qinchuan, Z. Applications of deep convolutional neural network in computer vision. *J. Data Acquis. Process.* **2016**, *31*, 1–17.
20. Fang, W.; Zhong, B.; Zhao, N.; Love, P.E.; Luo, H.; Xue, J.; Xu, S. A deep learning-based approach for mitigating falls from height with computer vision: Convolutional neural network. *Adv. Eng. Inform.* **2019**, *39*, 170–177. [CrossRef]
21. Dhillon, A.; Verma, G.K. Convolutional neural network: A review of models, methodologies and applications to object detection. *Prog. Artif. Intell.* **2020**, *9*, 85–112. [CrossRef]
22. Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349. [CrossRef]
23. Altan, A.; Karasu, S. Recognition of COVID-19 disease from X-ray images by hybrid model consisting of 2D curvelet transform, chaotic salp swarm algorithm and deep learning technique. *Chaos Solitons Fractals* **2020**, *140*, 110071. [CrossRef] [PubMed]
24. Bhattacharya, S.; Maddikunta, P.K.R.; Pham, Q.V.; Gadekallu, T.R.; Chowdhary, C.L.; Alazab, M.; Piran, M.J. Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey. *Sustain. Cities Soc.* **2021**, *65*, 102589. [CrossRef]
25. Pérez-Delgado, M.L.; Román-Gallego, J.Á. Medical image processing by swarm-based methods. In *Role of Data-Intensive Distributed Computing Systems in Designing Data Solutions*; EAI/Springer Innovations in Communication and Computing Series; Springer: New York, NY, USA, 2022.
26. Sezer, A.; Altan, A. Detection of solder paste defects with an optimization-based deep learning model using image processing techniques. *Solder. Surf. Mt. Technol.* **2021**, *33*, 291–298. [CrossRef]
27. Sezer, A.; Altan, A. Optimization of deep learning model parameters in classification of solder paste defects. In Proceedings of the 2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 11–13 June 2021; pp. 1–6.
28. Hemanth, D.J.; Deperlioglu, O.; Kose, U. An enhanced diabetic retinopathy detection and classification approach using deep convolutional neural network. *Neural Comput. Appl.* **2020**, *32*, 707–721. [CrossRef]
29. Akcay, S.; Kundegorski, M.E.; Willcocks, C.G.; Breckon, T.P. Using deep convolutional neural network architectures for object classification and detection within X-ray baggage security imagery. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2203–2215. [CrossRef]

30. Driss, S.B.; Soua, M.; Kachouri, R.; Akil, M. A comparison study between MLP and convolutional neural network models for character recognition. In *Real-Time Image and Video Processing 2017*; International Society for Optics and Photonics: San Diego, CA, USA, 2017; Volume 10223, p. 1022306.
31. Andreotti, F.; Carr, O.; Pimentel, M.A.; Mahdi, A.; De Vos, M. Comparing feature-based classifiers and convolutional neural networks to detect arrhythmia from short segments of ECG. In Proceedings of the 2017 Computing in Cardiology (CinC), Rennes, France, 24–27 September 2017; pp. 1–4.
32. Gan, K.; Xu, D.; Lin, Y.; Shen, Y.; Zhang, T.; Hu, K.; Zhou, K.; Bi, M.; Pan, L.; Wu, W.; et al. Artificial intelligence detection of distal radius fractures: A comparison between the convolutional neural network and professional assessments. *Acta Orthop.* **2019**, *90*, 394–400. [CrossRef]
33. Eiler, F.; Graf, S.; Dorner, W. Artificial intelligence and the automatic classification of historical photographs. In Proceedings of the Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality, Salamanca, Spain, 24–26 October 2018; pp. 852–856.
34. Bayr, U.; Puschmann, O. Automatic detection of woody vegetation in repeat landscape photographs using a convolutional neural network. *Ecol. Inform.* **2019**, *50*, 220–233. [CrossRef]
35. Mougiakakou, S.G.; Tsouchlaraki, A.L.; Cassios, C.; Nikita, K.S.; Matsopoulos, G.K.; Uzunoglu, N.K. SCAPEVIEWER: Preliminary results of a landscape perception classification system based on neural network technology. *Ecol. Eng.* **2005**, *24*, 5–15. [CrossRef]
36. Lienhart, R.W.; Hartmann, A. Classifying images on the web automatically. *J. Electron. Imaging* **2002**, *11*, 445–454. [CrossRef]
37. Szummer, M.; Picard, R.W. Indoor-outdoor image classification. In Proceedings of the 1998 IEEE International Workshop on Content-Based Access of Image and Video Database, Bombay, India, 3 January 1998; pp. 42–51.
38. Yiu, E.C. Image Classification Using Color Cues and Texture Orientation. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, UK, 1996.
39. Schettini, R.; Brambilla, C.; Cusano, C.; Ciocca, G. Automatic classification of digital photographs based on decision forests. *Int. J. Pattern Recognit. Artif. Intell.* **2004**, *18*, 819–845. [CrossRef]
40. Fernando, B.; Muselet, D.; Khan, R.; Tuytelaars, T. Color features for dating historical color images. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 2589–2593.
41. Palermo, F.; Hays, J.; Efros, A.A. Dating historical color images. In *European Conference on Computer Vision*; Springer: Heidelberg, Germany, 2012; pp. 499–512.
42. Martin, P.; Doucet, A.; Jurie, F. Dating color images with ordinal classification. In Proceedings of the International Conference on Multimedia Retrieval, Glasgow, UK, 1–4 April 2014; pp. 447–450.
43. Li, Y.; Hao, Z.; Lei, H. Survey of convolutional neural network. *J. Comput. Appl.* **2016**, *36*, 2508–2515.
44. Werbos, P.J. Backpropagation through time: What it does and how to do it. *Proc. IEEE* **1990**, *78*, 1550–1560. [CrossRef]
45. Hecht-Nielsen, R. Theory of the backpropagation neural network. In *Neural Networks for Perception*; Elsevier: Amsterdam, The Netherlands, 1992; pp. 65–93.
46. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
47. Müller, E.; Springstein, M.; Ewerth, R. "When was this picture taken?"—Image date estimation in the wild. In *European Conference on Information Retrieval*; Springer: New York, NY, USA, 2017; pp. 619–625.
48. Bradski, G. The openCV library. *Dr. Dobb's J. Softw. Tools Prof. Program.* **2000**, *25*, 120–123.
49. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in pytorch. In Proceedings of the NIPS 2017 Autodiff Workshop: The Future of Gradient-Based Machine Learning Software and Techniques, Long Beach, CA, USA, 9 December 2017.
50. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 8026–8037.
51. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
52. Dogo, E.; Afolabi, O.; Nwulu, N.; Twala, B.; Aigbavboa, C. A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks. In Proceedings of the 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS), Belgaum, India, 21–22 December 2018; pp. 92–99.
53. Zhao, H.; Gallo, O.; Frosio, I.; Kautz, J. Loss functions for image restoration with neural networks. *IEEE Trans. Comput. Imaging* **2016**, *3*, 47–57. [CrossRef]