A DEEP LEARNING APPROACH FOR PNEUMONIA DETECTION ON CHEST X-RAY

Manuel Vázquez Enríquez

Master’s Thesis presented to the Telecommunications Engineering School Master’s Degree in Telecommunications Engineering

Supervisor
Juan Carlos Burguillo Rial

2019
Abstract

In this master’s thesis we have implemented a deep learning architecture to diagnose cases of pneumonia in chest x-ray. The newly created dataset "RSNA pneumonia detection challenge dataset" created by the Radiological Society of North America has been used throughout the project. This dataset is the first and so far only dataset specialized in the detection of constitutive zones of pneumonia in chest x-ray and that will allow us to train neural networks specialized in tasks of classification and detection of objects. The implementation developed combines multiple convolutional neural networks of last generation. On the one hand, performing classification tasks we had integrated the convolutional neuronal networks Densenet, Inception-v3, Inception-Resnet-v2 and a Feed-Forward neural network built from scratch. On the other hand, focusing on the object detection task we had implemented the state-of-art Retinanet detector. At the same time, we have implemented combination and regularization techniques in order to maximize the performance of our implementation in both classification and detection tasks. The results obtained in the classification section reflect a significant improvement compared to reference work in the classification of cases of pneumonia such as CheXNet which obtained AUROC of 0.833, while our implementation provides AUROC score of 0.892. In the object detection section we get an AP@[.4:.75] score of 0.2496 which places our implementation in the top-2 into the leaderboard of the "RSNA pneumonia detection challenge".

Key words: RSNA, Pneumonia, Deep learning, Convolutional neural network, Classification, Object detection, Retinanet.
3.5 Metrics for detection and classification modules .......................... 30
  3.5.1 Metrics to evaluate classification task ............................... 30
  3.5.2 Metrics to evaluate object detection task ............................ 31

4 Preprocessing module ............................................................................ 33
  4.1 DICOM data extractor ................................................................. 33
  4.2 Black Border Removal ............................................................... 34
  4.3 Splitting Method ........................................................................ 36

5 Classification module ........................................................................... 39
  5.1 Stage 1: Multiclass-classification problem .................................. 40
     5.1.1 Architecture Design .......................................................... 40
     5.1.2 Experiments & Results ....................................................... 43
  5.2 Stage 2: Binary classification problem ....................................... 46
     5.2.1 Architecture design .......................................................... 47
     5.2.2 Experiments & Results ....................................................... 49

6 Object Detection Module .................................................................... 51
  6.1 Composition of the Retinanet architecture ................................. 52
  6.2 Architecture design of the object detection module ..................... 53
     6.2.1 Default retinanet architecture implementation .................... 53
     6.2.2 Anchor Boxes Density optimization .................................. 54
     6.2.3 Regularization Techniques ................................................ 55
     6.2.4 Score and NMS Filter of the output predictions .................... 56
     6.2.5 Ensembling methods ......................................................... 57
  6.3 Experiments & Results ............................................................... 58

7 Complete final architecture ................................................................. 61
  7.1 Architecture Design Summary .................................................... 61
  7.2 Enrichment object detection with global classification information .. 62
  7.3 Experiments & Results ............................................................... 63
  7.4 Evaluation in context ................................................................. 63
     7.4.1 Classification performance ................................................ 63
     7.4.2 Object detection performance .......................................... 65

8 Conclusions and future work ............................................................... 67
  8.1 Conclusions ................................................................................ 67
  8.2 Future work ................................................................................. 69

Bibliography ......................................................................................... 77
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Traditional automatic learning flow versus Deep Learning flow.</td>
<td>8</td>
</tr>
<tr>
<td>2.2</td>
<td>Biological Neuron versus Artificial Neural Network. Retrieved from [1].</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Feedforward structure with two hidden layers.</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>Numeric example of underfitting and overfitting cases.</td>
<td>14</td>
</tr>
<tr>
<td>2.5</td>
<td>Basic structure of a neural convolutional network. Retrieved from [2].</td>
<td>18</td>
</tr>
<tr>
<td>2.6</td>
<td>Imagenet ILSVRC challenge results since 2010 to 2017.</td>
<td>19</td>
</tr>
<tr>
<td>2.7</td>
<td>Accuracy vs computational complexity. Retrieved from [3].</td>
<td>20</td>
</tr>
<tr>
<td>2.8</td>
<td>Object detectors comparison table. Retrieved from [4].</td>
<td>22</td>
</tr>
<tr>
<td>2.9</td>
<td>Number of publications related to object detection. Retrieved from [5].</td>
<td>22</td>
</tr>
<tr>
<td>3.1</td>
<td>Two examples of Chest X-Rays into RSNA pneumonia challenge dataset.</td>
<td>24</td>
</tr>
<tr>
<td>3.2</td>
<td>Data distribution into RSNA Pneumonia Detection Challenge Competition Dataset.</td>
<td>27</td>
</tr>
<tr>
<td>3.3</td>
<td>General flowchart of the proposed implementation.</td>
<td>29</td>
</tr>
<tr>
<td>3.4</td>
<td>Workflow of the training process.</td>
<td>29</td>
</tr>
<tr>
<td>3.5</td>
<td>Workflow of the prediction process.</td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td>Flowchart with the actions applied in preprocessing module.</td>
<td>33</td>
</tr>
<tr>
<td>4.2</td>
<td>Dataset part of a sample DICOM file module.</td>
<td>35</td>
</tr>
<tr>
<td>4.3</td>
<td>Partial and final results of the black border removal algorithm.</td>
<td>36</td>
</tr>
<tr>
<td>4.4</td>
<td>Table of cluster centers and average silhouette before and after the black border removal algorithm.</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>Initial conceptual flowchart of the classification task module.</td>
<td>40</td>
</tr>
<tr>
<td>5.2</td>
<td>Architecture of Stage 1 of the Classification Module.</td>
<td>41</td>
</tr>
<tr>
<td>5.3</td>
<td>Parts of a single convolution model integrated in Stage 1.</td>
<td>42</td>
</tr>
<tr>
<td>5.4</td>
<td>Single Model Convolution Training Configuration Summary.</td>
<td>43</td>
</tr>
<tr>
<td>5.5</td>
<td>Training process results in stage 1 of the classification module.</td>
<td>44</td>
</tr>
<tr>
<td>5.6</td>
<td>Results of the predictions of each model in stage1 of the classification module.</td>
<td>45</td>
</tr>
<tr>
<td>5.7</td>
<td>Weights and results of the final predictions in stage1 of the classification module.</td>
<td>45</td>
</tr>
<tr>
<td>5.8</td>
<td>ROC Curve of the final predictions in stage1 of the classification module.</td>
<td>46</td>
</tr>
<tr>
<td>5.9</td>
<td>Initial Scheme of Stage 2 of the classification module.</td>
<td>46</td>
</tr>
<tr>
<td>5.10</td>
<td>Feedforward architecture of stage 2 of the classification module.</td>
<td>49</td>
</tr>
</tbody>
</table>
List of Figures

5.11 Results of the training process of the models of stage 2 of the classification module. .................. 50
5.12 ROC Curve and AUC-ROC score of the output predictions from stage 2 of the classification module. ........... 50

6.1 RetinaNet Detector Architecture. .............................. 52
6.2 Object detection Architecture ................................. 53
6.3 List of backones, input size and batch-size used. ............ 54
6.4 Distribution of the aspect ratio of the annotations in the bounding boxes. 55
6.5 Results of the optimization and increase in density of Anchor boxes. . . . 55
6.6 Representation of radiographic predictions before and after filtering. . . . 57
6.7 Results obtained with the anchor boxes density optimization and different Dropblock configurations. ...................... 59
6.8 Results obtained with the differentes backbone networks and input image size. ................................. 59
6.9 Final results obtained with the object detection module. ............... 60

7.1 Final results obtained in the task of detecting objects by the architecture built. ................................. 63
7.2 ROC Curve: Chexnet vs our implementation. .................. 64
7.3 RSNA Pneumonia Detection Challenge Leaderboard. .......... 65
7.4 Complete final architecture. ................................. 66
1 Introduction

1.1 Context

Pneumonia is a global health problem that does not understand of social or cultural strata, causing millions of deaths each year. In several conferences and articles it is called as "the silent killer", a nickname that reflects the little social and political awareness towards this disease that without receiving the same attention as other pathologies her number of affectionation year after year are forceful.

In developing countries, the affection of this lung disease makes it one of the most deadly among children under 5 years of age, causing 15% of the deaths recorded each year [6].

On the other hand, in Spain the people most at risk are those over 65. There are 162 incidences per 100,000 inhabitants per year, with a mortality rate of 5% for outpatients, 12% for hospital admissions and up to 40% for intensive care unit [7]. The latest report on "Deaths according to cause of death" published by the National Statistics Institute (INE) [8] in 2017 shows that respiratory system diseases were the third leading cause of death in all communities, with pneumonia being the most frequent cause of death. It also reveals that this pathology experienced a greater increase in 2017 in the number of deaths compared to previous years, increasing by 11.6% and standing at more than 10,200 deaths in that year.

Pneumonia is a disease of infectious origin that causes lung inflammation. The immune response of our body to this infection causes the lungs to fill with pus or other liquids reducing the ability to hold air, hence the patient may feel a choking sensation, cough and fever among other symptoms.

According to the regulations governing the diagnosis and treatment of pneumonia drawn up by the Spanish Society of Pneumology and Thoracic Surgery (SEPAR), the physician in the event of a possible case of pneumonia and after an initial clinical examination whose result is unspecific must obligatorily perform a chest X-ray (RxT) to establish a more accurate diagnosis [9].

The need for an early and accurate diagnosis to identify this pathology from the initial
clinical inspection and the interpretation of an X-ray is contrasted with the low repeatability of the cases, the difficulty of detecting it and differentiating it from other possible pulmonary anomalies. When interpreting an X-ray by a professional, his capacity and experience are key, but other factors such as fatigue, haste, subjectivity, discrepancies regarding to group consensus opinion and a long etcetera also have an impact. These factors result in significant variability among radiologists in detecting a case of pneumonia by radiography [10].

This variability in interpretation suggests the need for a paradigm shift from an exclusively manual interpretation of medical images by professionals to the introduction of CAD systems (computer-aided diagnosis) to support these professionals in interpreting and diagnosing while minimizing human errors. In addition, these systems will make it possible to train novice technicians with less experience and standardize the reading and interpretation of chest radiographic images.

Initially these CAD systems required a high manual development of selection of the main characteristics of the images and their interpretation, but now we are living the take-off of algorithms based on Deep Learning and more specifically based on convolutional networks (CNN) able to learn and extract characteristics and represent very complex nonlinear functions by themselves without human intervention based solely on input data after a supervised training process. This ability to learn by itself from neural networks opens up promising prospects for their application in the analysis and interpretation of radiographic images.

Various studies and works attest how Deep Learning algorithms have reached a performance equivalent or superior to human in various tasks and applications. A well-known case was documented on the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) when in 2015 a model outperformed the human-level performance in the task of classifying images [11].

Since then, models and algorithms based on Deep Learning in the field of computer vision have continued to improve, not only in the task of image classification but also in other tasks such as object detection or segmentation. The development of increasingly powerful GPUs has made these resource-intensive models possible. Finally, a third key factor in the success of the use of Deep Learning and at the same time a limiting factor the need for a very large volume of data with precise annotations with which to train the models in a supervised manner.

Deep Learning is an intensely data-hungry technology and although in the medical sector there has been a very large increase in Electronic Health Records (EHR) in recent years, the development of Deep Learning algorithms is still hampered by the availability of a dataset with precise annotations, a fact that is accentuated in the medical sector.

Concretely in the area that concerns us, that is, the diagnosis of cases of pneumonia from chest x-rays, we find that only a dataset called "ChestX-ray 14" created by the NIH (National Institute of Health) is available and with annotations focused solely on tasks of classification of pulmonary pathologies. This dataset gave rise to various works focused on the classification of pulmonary pathologies based on chest x-rays [12, 13, 14]. But
being the task of detection and location of anomalies impossible to develop due to the absence of a set of data with the necessary annotations for this type of task. Recently the Radiological Society of North America (RSNA) has published a dataset focused mainly on object detection and classification of pneumonia cases. The appearance of this new dataset gives us the possibility of developing more advanced Deep Learning models focused on the detection and diagnosis of signs of pneumonia. On the one hand improving the current classification models, and on the other opening up the possibility of applying or developing object detection architectures in order to detect and identify signs of pneumonia in radiographic images.

1.2 Objectives

The main objective of this work is to build a implementation based on Deep Learning that allows, from chest x-rays, to determine in a precise way the appearance of signs or anomalies capable of representing a case of pneumonia and to determine the location and framing of this anomaly. Providing a complete diagnostic tool for cases of pneumonia. We will work in depth on the recently created dataset provided by the radiological society of north america (RSNA). This dataset will provide us with the annotations data needed to train and evaluate our detection and classification architecture. Throughout the development of this work we will build a complete medical imaging architecture that will sequentially combine multiple imaging calculation techniques: preprocessing techniques, classification techniques and object detection techniques. The specific objectives are:


- Study of state-of-art computer vision techniques based on Deep Learning in classification and object detection tasks focused on both medical imaging and general use.

- Analysis and description of the dataset of the "RSNA pneumonia detection challenge" dataset and its use in the training and evaluation processes of our implementation.


- Implementation and evaluation of an object detection architecture based on deep neural networks.
1.3 Contributions

At the end of 2018, the radiology association RSNA made available to the research community a new set of data focused on the classification and detection of cases of pneumonia on chest x-rays.

The creation of a new dataset that increases the volume of medical imaging data accessible to the public is an important milestone that greatly benefits the research community and opens up a wide range of exciting and unknown possibilities: What added value can this new dataset bring to the research community?, will it allow us to build new models?, will it allow us to improve existing models?, and will it allow us to obtain results that go beyond the state of the art? Throughout this project, after extensive work on this dataset, we will discover the answers to these and other questions.

Our implementation combines techniques of two areas of computer vision, which are the tasks of classification and detection of objects. The development of the proposed implementation will allow us to know the possibilities that this new dataset can offer us in the diagnosis of cases of pneumonia in radiological images.

So, this work aims to offer a double contribution:

- On the one hand, to contribute to the scientific community providing a complete vision of the possibilities, limited or results of applying the most current models and techniques with this new dataset in the tasks of detection and classification.

- On the other hand, to contribute to the community of radiology professionals, proposing a tool to assist the professional that provides an accurate classification of the radiographic and information of the areas that show signs of pneumonia.

1.4 Thesis Outline

The present document has been structured in 8 chapters.

- In this first chapter, we have introduced the project offering a vision of the context and outlining the objectives and contributions of the project.

- In the second chapter, we will provide the reader with a background on the use of deep networks in the medical sector, and later, we introduce basic concepts on Deep Learning.

- Chapter 3 provides background information to enable a better understanding of the other chapters.

- Our implementation is structured in 3 modules: preprocessing module, classification module and object detection module. We will destine a chapter to each module covering from chapter 4 to 6.
1.4. Thesis Outline

- Chapter 7 will unify all the modules offering a complete view of the whole implementation and we will evaluate the results obtained with respect to the context.

- Finally, in chapter 8 we will dedicate it to conclusions drawn from the project and the introduction of possible future lines.
2 Background

2.1 Deep Learning in Medical Imaging

2.1.1 From traditional machine learning to Deep Learning

Nowadays, the need for an accurate diagnosis that reduces as much as possible the influence of human error has boosted the development and improvement of computer aided detection (CADe) and diagnosis (CADx) systems that provide support and assistance to medical professionals. Moving from a scenario where the medical professional diagnoses based solely on their knowledge and experience, to a scenario where the professional supports his diagnosis in systems trained in a particular problem.

In the development of these health care systems numerous papers support the distinction of two stages [15, 16, 17]:

- Pre Deep Learning era.
  In early 2000s the first computer-aided detection (CAD) is created. They were systems built using traditional methods of Machine Learning in which the features form the input were selected and extracted in a totally manual way and based on the criteria that the engineer considered appropriate. Correct selection and extraction of these features had a huge impact on system performance. Early reports warned that these systems generated more false positives and thus worse results than human reading, calling into question the benefit of such systems [18, 19].

- Post Deep Learning era.
  Deep Learning would bring a revolutionary breakthrough changing about how the model interacts and learns autonomously discovering complicated patterns and features directly from the input data. The selection and extraction of the feature set without human intervention provided numerous advances and the results improved enormously overcoming the limitations and poor results observed in the previous CAD systems.
Chapter 2. Background

The Figure 2.1 shows the paradigm shift. In the traditional machine learning, the Machine Learning algorithms are applied according to the features selected and extracted manually, while the Deep neural network learns how to extract the most representative or complex features and patterns on its own according to the input.

The research and development of Deep Learning has advanced giant steps, extending its application in a multitude of areas and varied problems, in which it has been positioned as the most powerful and referential models as for example in tasks of natural language processing, voice recognition, robotic, computer vision or recommendation systems.

In computer vision, an important milestone was reached when in 2012 on the ImageNet Large-Scale Visual Recognition Challenge a Deep Learning model based on convolutional networks surpassed the performance obtained with other Machine Learning techniques. This event demonstrated the enormous potential of Deep Learning in image processing tasks. Further developments and studies placed Deep Learning techniques based on convolutional networks as the *de facto* standard for a wide variety of computer vision task, such as image super-resolution reconstruction, object detection, image classification, face recognition or image segmentation.

The great impulse of Deep Learning has had its reflection in CAD processing systems the medical images provide a valuable diagnostic tool and assistance to the medical professional.

This momentum and success is underpinned by a number of factors:

- Increased availability of digitized data.
- Increased processing capacity, with increasingly powerful GPUs and increased access to computing power through IaaS services.
- A rapid development of algorithms and techniques that have significantly improved the technology initially proposed.
- Creation and distribution of user-friendly software frameworks such as Keras or Pytorch.
- Economic and investment drive in medical imaging market based on Deep Learning [20].
2.1. Deep Learning in Medical Imaging

2.1.2 Related Work

As indicated in the previous section, Deep Learning has experienced a strong momentum and enormous attention in recent years. As a result of this trend, numerous studies have emerged that analyze the influence of Deep Learning and mainly convolutional networks in the field of health and imaging medical diagnosis [21, 22, 15, 20, 23, 24]. All of them reflect the enormous interest and possibilities offered by the application of Deep Learning in medical image. Among these studies we highlight the paper [15] in which a complete compilation of more than 300 works is developed, whose common nexus is the application of Deep Learning in medical imaging.

In the detection and diagnosis of some of the pathologies with greater concentration of affection and mortality we find several works based entirely in the implementation of neural networks: Treating cancer [25, 26, 27], tumor [28, 29, 30] or cardiovascular problems [31, 32, 33].

If we focus on the pathology of the respiratory system that concerns us, the detection of cases of pneumonia, we find several works have incorporated the use of neural networks in their proposals for implementation [34, 35, 36, 37, 12, 13, 14]. Within the previously mentioned works we highlight a reference study in the diagnosis of cases of pneumonia on chest x-rays using Deep Learning called CheXNet [14]. In this work published by Stanford Machine Learning Group\(^1\), they implement a 121-layer Densely Connected Convolutional Neural Network and whose results will be used as a reference to contextualize our results in classifying pneumonia cases.

2.1.3 Challenges

Several studies [20, 22, 15, 24] collect a series of challenges for a successful implementation of Deep Learning for medical imaging tasks and act as a barrier that delays this process.

**Dataset:**

- The data is not complete homogeneous, it is obtained from different measuring devices or sensors with their corresponding calibration.

- The lack of a large training dataset with precise annotations. This circumstance is accentuated in several pathologies less common.

- Despite a drive to digitize medical information and implement PACS-like systems. There is no universal and standardized consensus of such systems and terminologies, which can lead to workflow incompatibilities.

- Dataset noisy and class imbalance. A small number of samples are available for

\(^1\)https://stanfordnlpgroup.github.io
some pathologies compared to other pathologies or cases without pathologies. This may result in an imbalanced dataset

- The complexity involved in interpreting medical imaging requires expert and experienced staff to create the ground-truth data, with the cost of expert staff time involved.

Technological advances:

- Need for high computing resources. Years ago this factor limited the implementation of Deep Learning systems even more. Today, advances in GPU hardware with increasing power and computational downtime techniques have partially overcome that limitation.

- Difficulty to train. Medical imaging is characterized by its complexity in the interpretation and in certain cases data limitation, an overtraining seeking to minimize the error can cause cases of overfitting of the system, losing generalization to new data. For this reason, optimisation and regularisation techniques have been developed and improved to minimise the effect of overtraining.

Ethical and legal:

- An implementation of Deep Learning can be seen as a black box where the complex relationships and interpretations deduced from input data are opaque producing a huge barrier of understanding these complex models by clinicians.

- Assignment of responsibilities in the event of a patient diagnostic error produced by the Deep Learning model.

- The creation and use of a dataset involves legal difficulties. It is necessary to completely anonymize the information and in the case of images this process can be difficult.

- A good reception and acceptance of such technology among medical professionals is necessary.

2.2 Introduction to Deep Learning

In this section we will briefly review the fundamentals of Deep Learning. We will begin by relating the principles on which it is based. Then we will offer a more practical view introducing the most relevant aspects of the training process and optimization of neural networks. Finally, we will focus on convolutional neural networks and provide an overview of the state of the art in the use of these networks in classification and object detection tasks.
2.2. Introduction to Deep Learning

Deep Learning is a branch of Machine Learning. The main and common feature is their ability to automatically learn representative features and functions from input information, which is called "representation learning" or "feature learning". Deep Learning carries out the training and with it the learning through multiple nodes or neurons distributed in multiple layers, simulating the functioning of the human nervous system, this similarity is shown in Figure 2.2. In the case of artificial neural networks (ANN) we start from a basic processing unit called node or neuron. Each neuron is itself a simple classifier model which generates an output as a function of the evidences of the previous layers. Each neuron performs three actions:

- **Linear operation.** A propagation function adds the signals coming from the previous layer with their respective weights and a bias value.

- **Non-linear operation.** On the propagation function applies an activation function which determines whether or not the output of the neuron is activated. There are several activation functions and depending on the space occupied by the node within the neural network it is convenient to use one or the other. The most common activation functions are: Sigmoid, Tanh, ReLU and Softmax.

- **Output.** The result of the activation function is located at the exit of the node and will propagate to all nodes located in the next layer.

These nodes or neurons are organized in layers and through the interaction with other neurons distributed in different layers allows to obtain more complex and hierarchical representations. Each layer essentially acts on the characteristic constructions of the layers prior to it generating a higher level feature construction.

Thus, in a neural network we encounter "N" nodes by layer and "C" different layers. In the simplest structure the nodes of the "C-1" layer are all connected to all the neurons of the "C" layer, this structure is called "feedforward" or "Multilayer Perceptrons" (MLP) and is the basic structure of Deep Learning. Figure 2.3 shows the feedforward structure
where the three main parts of a neural network are differentiated: Input Layer, Hidden Layers and Output Layer.

The number of input and output layers are fixed and their shape depends on the input dataset and the formulation of the problem or target function searched for respectively. But the size of hidden layers can be variable. If the model uses a single hidden layer it will be called "shallow neural network" and introducing more layers we will have a deeper model.

### 2.2.2 Types of Deep Learning algorithm

According to their purpose and the type of data used in the training process we find different types of algorithms, the main types are:

**Supervised algorithm.** They work with labeled data and the model is trained to minimize the cost function by comparing the prediction obtained by the model with the real value. Within this type we find: the feedforward neural network (FFNN) that we have commented previously, the Convolutional Neural Networks (CNN) used mainly in computer vision and Long Short Term Memory (LSTM) for applications based on time series data.
2.2. Introduction to Deep Learning

**Semi-supervised algorithm.** Only a small part of the samples have the annotations necessary to train the algorithm. This algorithm builds a self-learning scheme where the algorithm itself generates its own annotations [21]. Within this type we find Deep reinforcement learning (DRL) and Generative Adversarial Networks (GAN).

**Unsupervised algorithm.** The model discovers the structure or relationships of the input information without which it has annotations or labels. Auto Encoders (AE) and Restricted Boltzmann Machines (RBM) are two models that are located within this type and perform clustering or dimensionality reduction functions.

Throughout this project, the only models used will be supervised algorithms using mainly the CNN structure that we will explain in more detail in subsection 2.2.7.

### 2.2.3 How are supervised Deep Learning algorithms trained?

As we have already indicated, in Deep Learning it is not necessary to define a rule of prediction or selection of characteristics, but the model learns them by itself. The optimization algorithm used is called gradient descent and will seek to optimize the weights of the nodes until finding a local minimum of a function, i.e., minimize the cost function.

The descent gradient method consists of 3 steps:

- **Forward propagation.** Based on the current weights of each node a prediction is obtained.
- **Evaluation.** The obtained prediction is compared with the objective value, obtaining an estimated value from the error of the current weights configuration in the nodes.
- **Backward propagation.** This error is propagated in the opposite direction from the output to the input by applying slight changes in the weights proportional to the error and in the direction that minimizes the error.

These 3 steps will be repeated constantly until the end of the training process.

### 2.2.4 Improving Deep Neural Network: Cases of underfitting or overfitting

When training a Deep Learning model the main objective of our model is that you acquire the ability to generalize what you have learned, that is, from the data used in the
training phase, be able to obtain accurate predictions of samples not yet seen. The loss of generalizability of the model can occur in two directions [38]:

- Underfitting. The model has not been sufficiently trained or its structure is too simple to create an accurate representation of the input data. Showing a high bias, i.e. the difference between the predicted value and the actual value is high.

- Overfitting. The model has been overtrained and the nodes have ended up "memorizing" the input data. Showing a high variance, i.e. a significant difference between the error of the train set and the test set.

To conclude, if our model is correctly trained we must pay attention to the error obtained in both the train set and with the test set. Figure 2.4 shows a representative example of underfitting, overfitting and good fit cases.

In the first column we find a case of underfitting where the training error (bias) is high, we can conclude that has not yet converged on an optimal minimum. In the second column, we see that the error of bias is reduced but the difference between the error of the train set and the test-set is significant (high variance) which indicates a case of overfitting. In the third column, we see a reduced bias error but the variance error is also reduced, we would be faced with a well trained model.

In order to face the cases of overfitting or underfitting there are a series of strategies that allow to minimize them:

- In the case of underfitting. Possible strategies are to use a deeper network, use a more advanced optimization algorithm, improve its configuration, or change the neural network architecture.

- In the case of overfitting. It is usually related to a small dataset size [39], so it is reduced by increasing the number of samples in the train set or applying regularization techniques.

2.2.5 Improving Deep Neural Network (II): Optimization

As we have seen in the previous subsection, choosing the correct optimization configuration of our neural network will allow us to reduce the difference between the prediction
obtained and the correct value. In this subsection we will briefly show the different options we find when optimizing a neural network.

A. Gradient descent optimizer.
The function of the optimizer is to update the weights of the nodes in order to minimize the error of the cost function. At the time of selecting the optimizer we find several options: SGD, Adagrad, RMSprop or Adam.
We pay special attention to the Adam optimizer which combines the advantages of two optimization methods that are AdaGrad and RMSProp, obtaining a robust optimizer that adapts to a large number of problems [40].

B. Variants of Gradient Descent.
Depending on the amount of data used before updating the weights, we find 3 variants of gradient descent:

- Batch Gradient Descent. The cost function is not applied until all samples of the dataset have been passed, calculating the average error of the whole dataset to update each node by the backpropagation method.

- Stochastic Gradient Descent. The weights are updated with each new sample.

- Mini-batch Gradient descent. It is an intermediate point between the two previous ones. First the dataset is subdivided into mini-batches. Through the hyperparameter "batch-size" we configure how many samples will compose a mini-batch and the descending gradient method will calculate the average error and will update the weights once the samples that compose each mini-batch are passed.

The most common method is mini-batch gradient descent. This method combines the robustness of stochastic gradient descent and the efficiency of batch gradient descent. The "batch-size" parameter is the parameter with which it is configured.

C. Learning rate.
It is one of the hyperparameters that most affects the training process. A learning rate too small gives rise to a very slow convergence, while a learning rate that is too high cause divergence.
In addition, the optimal learning rate is not a constant value throughout the entire training process. It is important to start with a sufficiently high learning rate and reduce it as the model converges to global optimum.
To regulate and modify the learning value you can implement a learning rate scheduler. The general functioning of the various learning rate scheduler is based on the fact that after each epoch an action policy is applied in which it is defined in what circumstances
Chapter 2. Background

it is necessary or not to reduce the value of the learning rate and in what proportion it is reduced.

D. Weight Initialization Techniques - Transfer Learning.
When building a neural network the nodes, a previous step to train the model is to initialize the weights of the nodes. There are different options ranging from Random initialization to more sophisticated methods such as applying the He [11] or Xavier [41] initialization techniques.
Another widely used technique with satisfactory results is called Transfer Learning [42]. Transfer Learning consists of reusing the weights of a previously trained model against a very large dataset such as the Imagenet dataset [43] and with those reused weights start the training process being able to distinguish certain forms and characteristics inherited from the previous training in the early stages of training.

2.2.6 Improving Deep Neural Network (III): Regularization
To cope with an increase in variance when training as a result of overfitting we have several techniques to deal with it. Next, we will show a limited number of them corresponding to the techniques that we have used in the present work.

A. Data Augmentation.
A general rule when we are training any model is that the bigger the training dataset the better performance we will achieve. But sometimes the dataset is limited and it is not possible to obtain new samples with their corresponding annotations.
In order to increase the size of the training data set we have the possibility to apply certain transformations to the training data such as moving the image, zooming, applying some kind of distortion or noise... in this way we increase the number of training samples.

B. Dropout or Dropblock.
Dropout [44] is a regularization technique in which a number of nodes are randomly cancelled with each new sample in the training process. The ultimate goal is to reduce the interdependent learning amongst the neurons by facilitating generalization in the training process.
Dropout is an effective regularization method and provides important improvements to the training process, but its use is limited to fully connected layers. In convolutional networks where information is spatially correlated, although we apply dropout information continues to flow resulting in overfitting without dropout having any beneficial effect.
For convolutional networks there is a dropout variant called Dropblock [45]. Dropblock cancels blocks of nodes avoiding the propagation of some features maps towards later layers and with it achieving its function of regularization.

C. Early stopping.
Early stopping consists of monitoring a certain metric in the middle of the training
process. When the monitoring detects that the model is not improving or on the contrary that the variance increases, at that moment the training process stops automatically.

2.2.7 Convolutional Neural Network

In subsection 2.2.1 we introduced the basic scheme of a neural network showing the feedforward neural network structure where each node is connected to all the previous layer nodes. But this structure is not viable for working with images. The image is a multidimensional input where the pixels that compose it maintain a relation of local correlation with the neighboring pixels, if we used feedforward neural network the number of necessary parameters would make its training unfeasible. Therefore, to work with images arise convolutional neural networks [46]. Convolutional neural networks are a type of neural network specialized for working with images. These networks are inspired by the processing of Visual Cortex when the human brain processes an image. Each node covers a region of the image and together with the rest of the nodes the whole image is covered.

In a convolutional neural network 3 types of main layers are combined:

- Convolutional layer. Acts as a feature extractor for the input image. It applies a series of filters or kernels, which contain the trainable weights, and which run through the entire image executing convolution operations and generating a feature map. Each of the filters ends up specializing in detecting certain characteristics. In the first layers they detect simple patterns such as different lines or curves and in the later layers they detect more complex characteristics or patterns such as objects or shapes. After a convolutional layer an activation function is applied, most commonly using the rectified linear unit (ReLU).

- Pooling layer. This layer has the function of reducing the dimensionality of features maps. The most common operations are on the one hand to detect the maximum value of the subsample region discarding the rest to this operation or to calculate the average of the elements of each region [47].

- Fully-connected layer. This is one or more fully connected layers and is located close to the output. These layers learns the relationships between features maps and obtains output predictions that minimize the cost function.

As we can see in Figure 2.5 the different layers are combined following a logical sequence. The convolutional layer and pooling layer take care of extracting the features maps from the input image. And in the last layers are located the fully-connected layers that carry out the classification function, reaching a prediction from the feature map extracted from the image.
As indicated above, convolutional neural networks specialize in working with images, within this field we find different tasks: classification, detection and segmentation.

### 2.2.8 State-of-art in classification task

This task is well known in the computer vision community and consists of producing one or more descriptive labels for a given input image. Depending on the problem formulation we see that the classification task can be subdivided into:

- **Binary Classification.** The model will detect the presence or absence of a single class in the image.
- **Multi-label Classification.** The model will detect the presence or absence of each of the classes contained in a pre-established list of possible classes, which are not mutually exclusive, i.e., an image will be able to detect the presence of more than one class.
- **Multi-class Classification.** The model will only assign one class to the screen of a preset list of possible classes. The choice of a class is excluding and the image cannot be categorized with more than one class at a time.

**Evolution of architectures focused on the classification of images.**

The enormous evolution and improvement in performance obtained by Deep Learning models focused on image classification is closely related to the achievements obtained in the Large Scale Visual Recognition Challenge (ILSVRC) [43] that has taken place annually between 2010 and 2017. The objective of this challenge was to obtain the model that would obtain the best performance using only the Imagenet dataset. During the years that this challenge took place it has hosted the great advances of Deep Learning in classification tasks, in Figure 2.6 we gather the winning architectures every year. In 2010 and 2011, traditional machine learning was still used using Feature engineering with a result 5 times higher than the estimated human error, which was defined as a 5%
2.2. Introduction to Deep Learning

Although in 2010 Yann Lecun had already created the first convolutional neural network \[46\], it was not until 2012 that a convolutional model was presented in ILSVRC. The model called Alexnet \[48\] was an important milestone in the field of computer vision. On the one hand, it halved the winning result for 2011 and, on the other, the scientific community confirmed the substantial improvement involved in using convolutional neural networks for image processing tasks. Since then, all the winning models of this challenge have been based on this type of neural network.

Thus in 2013 LeNet-5 and in 2014 the model GoogleNet or Inception V1 \[49\] won this competition, both presented deeper and deeper networks with more and more layers achieving better results with the Imagenet dataset.

In 2015, another important milestone was reached. The ResNet model \[50\] surpasses the estimated barrier of human error, which meant that the Deep Learning model was able to improve the performance of a human to classify images. Thereafter, in 2016 DenseNet \[51\] achieved a better result that Resnet optimizes the early gradient flow by implementing a densely connected layers.

In 2017, in the year in which the challenge ends, the winning model was the ScENET or Squeeze-and-Excitation Networks architecture \[52\].

From 2017 until today, convolutional models focused on classification tasks have continued to improve year after year by introducing more layers to previous models, by combining models, by introducing ingenious techniques that improved neural network optimization or by directly creating a model from scratch fully optimized to the problem by reinforcement learning search method such as NasNet \[53\].

Figure 2.7 shows a compilation of the extensive range of models based on convolutional neural networks, where it is observed that as the performance of older models are significantly worse than the performance obtained with more current models such as SENet or NASNet networks.
2.2.9 State-of-art in object detection task

The object detection task allows you to identify one or more objects in an image, providing information about their location in the image and their confidence score.

In early stages, before 2012, object detectors followed a traditional scheme using handcrafted Machine Learning features. Viola Jones Detector [54] can be considered the first object detector, which will be followed by other implementations such as HOG [55] (Histogram of Oriented Gradients) Detector and DPM [56] (Deformable Part-based model). DPM was considered for a few years the reference algorithm, until in 2012 began to emerge the first object detectors based on Deep Learning surpassing the performance of previous approaches.

Now in the hands of Deep Learning, object detectors have made great progress in terms of accuracy, speed and efficiency. With respect to how the detector is built we can differentiate two main frameworks or implementations: Two-Step Detectors and Single-Step Detectors [5].

A. Two-Step Detector.

As the name implies, these detectors act in two steps:

- Step 1 or Region Proposal Step: A set of regions with a high probability of containing a searched object is generated.
- Step 2 or Object Detection Step: On the proposed regions the bounding boxes are defined and the detection obtained is classified.
2.2. Introduction to Deep Learning

The two-step detector frameworks best known are:

- **R-CNN [57]**: First, the network selectively searches for a set of regions in the image that might belong to a proposed object. Secondly, each proposal is introduced into a CNN model to extract its features. Finally, two networks are trained: a bounding-box regression to learn corrections in the bounding box provided and a SVM classifier that will get the label and confidence score assigned to that bounding box.

- **Fast R-CNN [58]**: Improves the R-CNN implementation. Instead of training as many convolutional models as regions proposed, a single CNN is used over the entire image, generating a convolutional feature map. This feature map generates the proposal regions by a method called Region of Interest (RoI). These regions go through a pooling layer that maintains a fixed size for all proposed regions that will feed the fully connected layer, followed by a softmax layer to predict the class and bounding-box regression to define the resulting bounding box.

- **Faster R-CNN [59]**: The selective search proposed in the previous algorithms causes the network to be slow. Faster R-CNN introduces a key novelty in the Fast R-CNN architecture. Instead of using the selective search algorithm it will use a different network called Region Proposal Network (RPN) that will generate more efficiently and accurately the proposal regions of the image.

**B. Single-Step Detector.**

Unlike two-step detectors, these detectors directly predict the bounding boxes and confidence scores for the image without any intermediate step. The most popular one-step detector frameworks are:

- **YOLO (You Only Look Once) [60]**: Applies only once a convolutional network to predict both bounding boxes and their score confidence. To do this, it divides the image into 7x7 grids and predicts X bounding boxes with their corresponding score confidence and finally filters the bounding boxes whose score confidence does not exceed a certain threshold. Just passing the image once over the CNN network will detect a series of bounding boxes and score confidences. This makes it a very fast implementation. Several improvements have been introduced in later versions. In versions V2 [61] and V3 [62] the prediction accuracy is improved without significantly increasing the time required.

- **SSD (Single shot detection) [63]**. When an image passes through a convolutional network its scale is reduced after each layer of the network. SSD will act not only on the output of the convolutional network but also on different intermediate convolutional layers. This technique allows you to detect objects of different scales, improving their detection, especially when faced with small objects.
Chapter 2. Background

Figure 2.8: Object detectors comparison table. Retrieved from [4].

Retinanet[4]. One-step detectors stand out for their speed and resource efficiency compared to two-step detectors, but provided a prediction with lower accuracy. The creators of Retinanet analyzed the reason for the loss of precision suffered by one-step detectors and came to the conclusion that the main reason was the foreground-background class imbalance during the training process. The solution to this problem translates into a new loss function called "focal loss". Focal loss weighs samples by focusing on "difficult" samples and reducing the influence of clearly negative samples. In addition, it incorporates the FPN implemented over the convolutional network that by default is the Resnet architecture. The combination of using a Resnet-FPN backbone and focal loss makes the Retinanet detector surpass the two-step detectors in accuracy while maintaining its superiority in terms of speed and efficiency typical of single-step detectors.

Figure 2.8 shows the performance of various two-step versus one-step detectors. It can be seen that Retinanet obtains the best values in each of the metrics.

There is currently a rebound in the development and improvement of object detectors, integrating into numerous real-world applications such as face detectors, autonomous car or medical diagnosis. This growing interest is reflected in the constant annual growth in the number of publications as reflected in Figure 2.9.

Figure 2.9: Number of publications related to object detection. Retrieved from [5].
In this chapter we will cover general aspects of our implementation. In the first instance and since during the project we refer repeatedly to the presence and absence of representative areas of pneumonia in Chest X-Ray images, we will dedicate a section to introduce the reader to the understanding and visual appreciation of the presence of pneumonia. After that, we will take a look at the generic and global topics which are present throughout the following chapters. In these topics we will introduce the dataset used, the general structure of the proposed architecture, the strategy used when training and evaluating each of the Deep Learning models used and finally the metrics used.

3.1 Basic of interpretation case of pneumonia into CRX

Throughout the project we repeatedly refer to the presence or absence of signs constituting pneumonia in an image. But what do we mean by signs of pneumonia? How do you see them visually in an image?

An x-ray is a grayscale image where the lighter areas are areas that have absorbed more radiation allowing bones to be identified. At the other extreme the areas that absorb less radiation are areas full of air, therefore, the air housed in the lungs will have its representation in the x-ray with a very dark color close to black. Finally, between both extremes with tonal variations within the gray scale can be appreciated tissues and fluids.

In Figure 3.1 we show two examples that belong to the dataset used, where you can see the variation in the gray scale described above.

Pneumonia provokes in the patient that his lungs fill with pus or other fluids reducing his capacity to hold air. Regarding its interpretation in an X-ray, instead of seeing an air-filled lung with a tone close to black in all its extension, certain opacities will appear inside the lung. The term opacity in the interpretation of x-rays is defined textually as: "any area that preferentially attenuates the x-ray beam and therefore appears more opaque than the surrounding area" [64].

When interpreting a chest x-ray, a first approximation is to associate the appearance of
certain opaque areas in the area of the lungs with a possible case of pneumonia resulting from a lung infection.

In Figure 3.1 we can see that in the image on the left-hand side there are no signs of pneumonia, i.e. there is no opacity inside the lungs, but rather there are lungs filled with air uniformly. On the other hand, the image on the right shows signs of opacity inside the lung, which at first sight would be more likely to correlate with a case of pneumonia. The examples shown in Figure 3.1 have been chosen from the total dataset radiographs because they clearly show a patient without pneumonia versus one with that condition, where even an untrained eye could try to fill an initial diagnosis. But in the vast majority of x-rays the diagnosis is much more complicated and complex: images of poorer quality and with a smaller tonal variation, the appearance of other lung diseases that can affect the diagnosis such as masses, nodules, lobar collapse and so on ... can be easily misleading or misdiagnoses.

Another salientable aspect is that delimiting the bounding boxes that delimit the opacity when faced with a case of pneumonia is very complicated and complex to determine exactly the limits and extension that this opacity covers.

3.2 "RSNA Pneumonia Detection Challenge Competition"

Dataset

In this section we will focus on this dataset, which is the main dataset from which we will train and evaluate our implementation proposal.

This dataset is the result of the work during 6 months of 18 radiologists with an average experience of 10.6 years, coordinated by the RSNA association. Recently, it has been used in a competition of the Kaggle platform called RSNA Pneumonia Detection Challenge. After this competition this dataset has been maintained public and available for its use by the scientific community [65].

It is a collection of 25684 samples taken from a larger dataset of 112000 samples created
3.2. "RSNA Pneumonia Detection Challenge Competition" Dataset

by the National Institutes of Health (NIH). Although the RSNA dataset is a subset of the NIH dataset, there are numerous differences that make it an improved dataset focused on pneumonia cases, providing accurate and useful information to be used in classification and detection tasks.

In the following, we list the differences between the two datasets and give a brief introduction to what a file in DICOM format consists of.

We end this section with a brief exploratory data analysis (EDA) of the RSNA dataset, where we will describe the elements and parameters contained in the dataset and the general distribution of the samples.

3.2.1 Improving the National Institutes of Health Chest Radiograph Dataset

As indicated above, the RSNA Pneumonia Detection Challenge Competition Dataset is a subset of the National Institutes of Health Chest Radiograph Dataset. Although they share common samples this new dataset introduced numerous differences [65]:

- Sample format. The NIH dataset only provides samples in .png image format. In contrast, the RSNA dataset offers the samples in a specific format used in the medical field called DICOM (Digital Imaging and Communication On Medicine). This change of format allows to insert in the dataset files the radiographic image in addition to other information of the patient.

- Reduction of the total number of categories and improvement of their precision. The NIH dataset proposes up to 14 pulmonary pathology classes: Atelectasis, Cardiomegaly, Consolidation, Edema, Effusion, Emphysema, Fibrosis, Hernia, Infiltration, Mass, Nodule, Pleural Thickening, Pneumonia and Pneumothorax with an estimated accuracy of 90%.

  In contrast, the RSNA dataset specializes only in cases of pneumonia. It unifies the categories of consolidation, infiltration and pneumonia into a single one because the three classes constitute a case of pneumonia and improves the precision with which the samples have been categorized.

  The result is a more accurate dataset categorized in only 3 classes: Normal, lung-opacity and not-normal.

- From multi-label classification to multi-class classification. The nature of the NIH dataset is oriented to the formulation of a multi-label classification problem, whereas RSNA dataset formulates a multi-class classification problem. Therefore, in the RSNA dataset the samples will be categorized with only one class.

- Aggregate annotations focused on object detection. One of the most important and differentiating characteristics of the RSNA dataset is that this dataset provides the bounding boxes of the areas of the image where the opacity associated with a
pneumonia case is appreciated. This peculiarity makes it the first and to this day the only dataset available with which to train object detection models in cases of pneumonia.

- Elimination of the cases where serious deficiencies in the radiography and outliers cases are observed.

### 3.2.2 Exploratory Data Analysis

Once we have seen the differences and new features provided by the RSNA Pneumonia Detection Challenge Competition Dataset, we will describe what elements it is composed of and how is the distribution of its samples. The dataset is composed of chest X-rays in DICOM format. DICOM is a special format widely used in the medical field that includes the radiographic image and a series of patient data. Along with these files are included two excel files with the annotations of bounding boxes and classification of each of the samples. Thus, for each sample we will have the following information:

- Patient ID. Unique identifier of the sample. It is the link between the x-ray images and the annotations for each of the samples.

- Radiographic image. Image rescaled to 8-bit encoding (256 grayscales) and with fixed dimensions of 1024 x 1024.

- Sex of the patient. Categorical variable.

- Patient’s age. Numerical variable.

- View. Provides information on how the x-ray was taken. In PA view the x-ray passes from posterior of body to anterior and vice versa in the case of the AP view. Preferably the PA view is used, but for very sick people who cannot maintain their upright position the AP view is used.

- Classes. Categorize the image in one of the 3 possible classes that are:
  - Normal: Sample without any trait of pneumonia or any other pulmonary pathology.
  - Lung opacity: Sample constitutive of a case of pneumonia.
  - Not-Normal: Sample that does not constitute a case of pneumonia, but which reflects any other pathology.

- Target. It reduces the previous categorization in Binary x-ray category where 0 means that the image is not a case of pneumonia and 1 if it is a case of pneumonia.
3.2. "RSNA Pneumonia Detection Challenge Competition" Dataset

Bounding boxes. List of bounding boxes which frame areas which constitute pneumonia. In samples with pneumonia (Target = 1) the samples can have 1 or more bounding boxes. Each bounding box will be defined by 5 elements which are $x_1, y_1, x_2, y_2$ and confident score. The first 4 elements are the coordinates that delimit the box detected and the last element, confidence score, defines the probability that this box framed an area of pneumonia.

Next, in Figure 3.2 we show how to distribute the information that constitutes the dataset for each parameter previously mentioned, in the case of bounding boxes we show the distribution of the centers of each box. Observing this distribution we can extract the following information:

- As can be expected, there is a direct correlation between the Target and Class parameters. The only difference is that Target is a binary classification and we would find an unbalanced distribution between both possibilities, as opposed to Class where the category "No" is subdivided into the categories "Normal" and "No Lung Opacity / Not normal".
- In the parameters Sex and View we see a slightly higher distribution of men and catches with PA view. But the difference is not significant.
- In the parameter of age, we see that the dataset corresponds to an adult population reaching old age, with few samples in the earliest ages.
• With respect to the bounding box centers. We can see two clearly differentiated groups which correspond to each of the lungs. But some centres are dispersed and distant from these two main clusters. Reviewing the images this variation is not due to false positives, but to the fact that some x-rays do not occupy all the space and are displaced. This anomaly will be analysed and dealt with in more detail in chapter 4 of the preprocessing module.

3.3 General structure of the proposed implementation

Figure 3.3 shows the general flowchart of the proposed implementation. It can be appreciated that in this architecture 3 main modules are differentiated. Each one of them with a specific functionality and task that will act and interconnect in a sequential way, following the following order:

1. Preprocessing Module. This module includes the preprocessing actions that we will apply to each and every input sample with which we feed the models during the training and evaluation processes.

2. Classification Module. Module specialized exclusively in image classification. The output of the module shall consist of a single probability value between 0 and 1 of the presence or non-presence of pneumonia globally, i.e. as a function of the entire image as a whole. In the first place, this module will allow us to globally categorize the sample in 2 classes: image with case of pneumonia and image free of any trace of pneumonia and secondly its output can directly influence the predictions obtained in the detection module.

3. Object detection Module. Module specialized in the task of object detection. The result of this module will be a list of vectors where each one of them defines a bounding box detected. Each bounding box will be defined by 5 elements which are x1,y1, x2, y2 and confident score. The first 4 elements make up the coordinates that delimit the box detected and the last element, confidence score, defines the probability that this box frames an area of pneumonia.

An exclusive chapter will be dedicated to each of the modules mentioned above, where we will describe them in more detail.

3.4 Training and prediction strategies

In this section we will give some brushstrokes of the strategy followed when training the multiple models that are part of both the classification module and the detection module. There are three basic starting points:
3.4. Training and prediction strategies

- The models trained will be mainly convolutional neural networks which require supervised learning.

- In the modules of classification and detection we have sought to maximize the objective metric for which we have applied techniques of combination of multiple predictions for each model and more than one model in a weighted way.

- The dataset used is composed of DICOM format files and their corresponding annotations will be divided into 3 portions: train, validation and test. In the training process we will use only the training and validation portions. Test set will be reserved to evaluate the trained models and in the processes of combination of predictions and models.

Figure 3.4 shows the workflow of the training process. In this process only the training and validation subsets act. After each epoch the current model will be evaluated using the validation set looking for optimal values of bias and variance. In both the training and detection modules we will train multiple models.

Figure 3.5 shows a simplification of the workflow that takes place in the prediction process. Once we have all the models trained we apply two techniques:

- Test-time augmentation (TTA). This technique consists of the output prediction of a model being the result of the unweighted combination of multiple predictions calculated by said model with different versions of the same input samples, that is, between each prediction some type of change or transformation is applied to the input samples (data augmentation).
Chapter 3. Preliminary considerations

The application of the test-time augmentation technique provides important improvements in the predictions of each of the trained models. In the early work of Deep Learning the use of data augmentation was applied only in the process of training on a training set as a regularization technique. But recently, several works have emerged that begin to introduce this technique and empirically demonstrating the improvements it provides, obtaining more robust precisions [66, 67, 68].

- Multi-model ensemble. Combining the predictions of multiple models allows us to obtain a more accurate and robust prediction. The ensembling method used is called "Bagging" [69] and consists of combining the predictions of each of the models in a weighted way. To select the weights applied to each prediction we will use a "grid search algorithm" which consists of calculating the performance with all possible combinations of weights and record the combination with which the target metric maxime against the test set.

3.5 Metrics for detection and classification modules

This section will describe the main metrics used to evaluate the performance of both the classification and detection module.

In the research community there is an extensive list of possible metrics, which may be used to measure the performances of an architecture or implementation. Although in the training process we compute the error and accuracy against the cost function, it is normal that in order to evaluate the performance obtained with guarantees, another more advanced metric is used. Depending on the application and the problem, a mistake or another can significantly impact your performance [39] and using the most widely used metric for that type of problem allows you to compare with other implementations using the same metric.

3.5.1 Metrics to evaluate classification task

Several reviews in the field of radiology [70, 71] and works of classification of pulmonary pathologies [12, 14], similar to ours work, put special emphasis on the use of the ROC Curve and the AUC-ROC score as a metric for this kind of problems. The ROC curve
3.5. Metrics for detection and classification modules

confronts in a graph on the one hand in the vertical axis the proportion of positive correctly identified (Sensitivity) and on the other hand, in the horizontal axis, the proportion of actual negative correctly identified (1-Specificity) under several classification thresholds. Finally, the area under the curve (AUC) measures the entire area below the ROC curve, providing a quantification of the performance given by the model to evaluate across all classification thresholds.

3.5.2 Metrics to evaluate object detection task

In the area of object detection there is a great variety of metrics, but all of them follow a very similar calculation procedure differentiated by a slight modification that we will explain later.

First of all, we calculate the value of IoU (Intersection over Union) between the predicted bounding boxes and the real bounding box by equation 3.1

\[
\text{IoU}(A, B) = \frac{A \cup B}{A \cap B}
\]  

(3.1)

Once we have calculated all the IoU values for each predicted bounding box, we define a threshold with which we will evaluate the obtained IoU values and we will obtain the number of False Positives (FP), True Positives (TP) and False Negatives (FN) and finally calculating the Average Precision (AP) by the equation 3.2.

\[
AP = \frac{TP}{TP + FP + FN}
\]  

(3.2)

Keeping the previous steps we find different variants, the most important and used are:

- **Pascal VOC Challenge metric** [72]: AP50. A single threshold of 0.5 is defined.
- **COCO Object Detection Challenge metric** [73]: AP@[0.5:0.95]. A range of thresholds ranging from 0.5 to 0.95 with a step size of 0.05 is defined. The AP score is calculated for all thresholds and finally an average is computed.
- **RSNA Pneumonia Detection Challenge metric**: AP@[0.4:0.75]. Use a range of thresholds from 0.4 to 0.75 with a step size of 0.05.

In the present work we will use at all times the RSNA Pneumonia Detection metric, which will simplify its nomenclature to AP\textsubscript{RSNA}. We choose this metric over other metrics exposed because this metric is the recommended and used metric in the RSNA Pneumonia Detection Challenge, which will allow us to compare our results with the results obtained with the other participants.

\[\text{https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/overview/evaluation}\]
In this chapter we will focus on the actions applied on the dataset samples as a previous step to feed the models of the classification and object detection modules both in the training and prediction process. In Figure 4.1 we group in a flowchart all the actions applied in this module and that we will explain in detail in the following sections.

### 4.1 DICOM data extractor

The samples that compose the dataset are presented in DICOM format. DICOM (Digital Imaging and Communication on Medicine) is designed and created to be used in PAC systems (Picture Archiving and Communication system) in the field of medicine. A DICOM file is structured in two parts file [74]:

- **Header.** It has two parts, a 128-byte preamble with information about the specific application and a constant prefix of 4 bytes with the letters 'D', 'I', 'C' and 'M'. This information is used in tasks of management and organization of the PAC systems.

![Flowchart with the actions applied in preprocessing module.](image)
• Dataset. It’s a listing of data known as Data Elements. Each data element provides information associated with the patient. Figure 4.2 shows an example of a "dataset" field of a DICOM file of the dataset used. As you can see, the information is anonymized, preventing patient identification.

After analyzing the structure of which the input files are composed. An algorithm has been created that will be applied with each of our samples to extract the relevant information from the DICOM file. In its development we have relied on the Python library with MIT-based license pydicom¹. The main steps that take place:

• Extract the information we will use in the system, i.e. the patientID, patient’s sex, patient’s age, view position and pixel data fields.

• Encoding string categorical fields Patient’s sex and view position to numeric value.

• Convert the 1024x1024 array extracted from pixel data into a .png file using the patientID as the file name.

4.2 Black Border Removal

Analyzing how the centres of the bounding boxes are distributed throughout the 1024x1024 space of the image we saw in subsection 3.2.2, it can be seen that some centres are located at the ends of the image, separating themselves from the nucleus of the clusters. The visualization of these cases confirms us that not all the images occupy all the possible space, finding images with very pronounced black margins as can be seen in the first image of the Figure 4.3. For this reason we have created an algorithm that minimizes the black margins of the X-rays without varying in any case in the aspect ratio or size of the resulting image.

IMPLEMENTATION.
The algorithm performs in four steps:

1. Search for the points that delimit the useful information, i.e. the radiography. And we define the rectangle that best contains the radiography without altering the inclination or other characteristics of the original radiography.

2. Approximate the result of step 1 to a square with aspect ratio 1:1. To do this, we add the minimum necessary margins and we keep the X-ray in the central position of the image.

4.2. Black Border Removal

Figure 4.2: Dataset part of a sample DICOM file module.

3. Resize the image to the same size as the original image, i.e. 1024x1024. For this we have used the Lanczos interpolation over 8x8 pixel neighborhood through the functionality of the opencv library "cv2.INTER_LANCZOS4". The choice of interpolation is based on the prioritization of image performance in terms of preservation and reduction of aliasing artifact, as opposed to the time and complexity of interpolation computation [75].

4. In previous steps it has been possible to move and enlarge the constituent zone of pneumonia, we adjust the boundary boxes in the annotation files according to the changes applied.

EVALUATION.

In the distribution of bounding box centres, two main clusters are clearly visible. To evaluate the improvement introduced with the proposed algorithm, we have evaluated how well the centers are grouped based on these two clusters applied Silhouette Analysis [76] on KMeans Clustering, that is, to evaluate the improvement by measuring the average silhouette coefficient of all centers.

\footnote{https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html}
Chapter 4. Preprocessing module

Figure 4.3: Partial and final results of the black border removal algorithm.

Figure 4.4: Table of cluster centers and average silhouette before and after the black border removal algorithm.

Figure 4.4 shows the centers of clustering and the average silhouette measure of both the original raw image and the image after applying the black border Removal algorithm. There is a slight improvement in the quality of clustering achieved once we remove the black borders in the input images.

4.3 Splitting Method

As we have seen in the previous chapter the dataset consists of a total of 25684 samples in DICOM format and a file with ground-truth annotations for each sample. Following the training strategy defined in section 3.4, we must group the samples with their corresponding annotations into 3 portions: training set, validation set and test set. Although this task may seem trivial, not performing a correct partition of the dataset will have negative effects that will affect continuously throughout the whole process of both training and evaluation. In order to reduce the effects of underfitting and overfitting, the partitioning method must ensure that each portion is a good representative of the whole, i.e. that each portion maintains a similar percentage of each class.

The splitting method we have used is based on the StratifiedShuffleSplit\(^3\) functionality built into the BSD-licensed\(^4\) machine learning scikit-learn library.

The percentage of the entire dataset in each serving has been established as follows:

- Train Set: 80%.
- Validation Set: 10%

---

\(^4\)Freeware License Berkeley Software Distribution.
• Test Set : 10%.

And it has been prioritized to maintain in each portion a similar percentage of samples of each class within the category "Class" of the file of annotations. This category as we saw in the subsection 3.2.2, groups the samples around 3 classes: normal, Lung opacity and non-normal. Therefore, given the dataset annotation file we will create 3 different annotation files, one for each portion with a specific role: training, validation and test. Each one of the subsets will maintain all the columns, being the values of the patient_ID column the link between the annotation files and the DICOM files.
In this chapter we will focus on the development and evaluation of the performance of the classification module. This module as its name indicates specializes in the task of classification providing a global prediction of all the radiography as a whole. Given a sample in DICOM format we will obtain the probability of presence or absence of pneumonia by globally categorizing the radiography.

When designing this module we started from the following premises:

- The objective of the module is to provide a binary classification of presence-absence of pneumonia within a chest x-ray.

- From the analysis of the distribution of the dataset samples (subsection 3.2.2) it is observed that the categorization based on the binary parameter "Target" (presence and absence of pneumonia) shows that the dataset has a significantly higher percentage of negatives than positives. Positive cases occupy only 22% of the total samples. On the other hand, using the categorization of the radiography based on the "Class" parameter composed by 3 classes, the distribution of the classes is correctly balanced, which has a positive impact on the training process of the models.

- From DICOM files we obtain both the chest x-ray image and additional patient information. This information could provide added value in the overall image classification process by improving the performance of the proposed classification module.

For this reason, our implementation proposal is based on 2 stages. In Figure 5.1 we show through a flowchart the sequence of stages that take place for each input sample:

1. Stage 1. We formulate the problem as a multi-class classification task. In this stage we will only work with the image of the radiography and by a convolutional neural
networks we will assign to this radiography one of the 3 possible classes: normal, lung opacity and no-normal. The output of this stage will be the probability for each of the classes.

2. Stage 2. On the predictions obtained in the previous stage we explored the inclusion of a series of additional patient information incorporated into DICOM files. To do this we have built from scratch a neural network based on a FFNN (FeedForward Neural Network) topology. We formulate the problem as a binary classification task of presence or absence of pneumonia.

### 5.1 Stage 1: Multiclass-classification problem

In this first stage of the classification module we formulate the problem as a multi-class classification task, i.e. given an input sample x (a frontal-view chest Xray image) should be classified to one and just one from the predefined classes $y \in \{\text{Normal, Lung opacity, No-normal}\}$. 

In order to maximize the quality of the prediction we have followed the guidelines described in section 3.4. We have built an architecture that combines the predictions of 3 convolutional neural networks in a weighted way. Predictions from these models have previously been obtained by applying the test-time augmentation technique. And finally, each of the models has been individually trained with the same data set (train set and validation set).

In the following, we will describe the designed architecture. And finally, in subsection 5.1.2 lists all the results obtained in both the training and prediction processes.

#### 5.1.1 Architecture Design

As we have already mentioned, we have trained 3 convolutional models. We have implemented three state-of-the-art networks called Densenet [51], Inceptionv4 [49] and Inception-resnetv2 [77]. All of them are widely known and used in the field of image classification.
5.1. Stage 1: Multiclass-classification problem

Each of these models has been trained individually to detect and identify features associated with cases of pneumonia. But training the models from scratch is very costly in terms of the time and quantity of samples required to train satisfactorily. Therefore, the training will start from the implantation of neural networks that have been pre-trained in classification problems against the Imagenet dataset. This technique is called Transfer Learning, which we discussed in subsection 2.2.5. Once each model has been trained, we have built an architecture in which all the trained models are integrated. This architecture will allow to combine multiple predictions for each one of the models and at the same time to combine the predictions of each one of the models in a weighted way obtaining a single prediction.

Figure 5.2 shows the structure of the architecture designed for stage 1 of the classification module.

Next, we will delve into the necessary steps to implement and adapt each of the convolutional neural networks to the dataset and type of problem posed. On the other hand, we will describe the algorithms applied in the prediction process.

**Single model architecture.**

To implement each of the models we will use both the definition and the pretrained weights that the Application module\(^1\) of the Keras Deep Learning library provides us.

In the case of the Densenet architecture, multiple versions of this architecture are available, differentiating between them by their depth, that is to say, the number of layers used. In our case we will use a depth of 121 layers. The reason for this choice is that Densenet-121 is the convolutional neural network used in the CheXNet project [14], a reference project in the classification of pulmonary pathologies on chest x-rays.

When implementing convolutional models, the Keras library provides the definition and weights for a common and generical architecture. In order to adapt it to our problem we have to incorporate an input stage and an output stage according to the formulation and requirements of the problem.

Figure 5.3 shows the steps we have incorporated before and after each convolutional neural network:

\(^1\)https://keras.io/applications
Chapter 5. Classification module

Figure 5.3: Parts of a single convolution model integrated in Stage 1.

- Preparation: In this part we adapt the original size of the image to a more manageable size, according to the input expected by the pretrained models. Thus, for the Densenet121 model we have reduced the image to 224x244 and for the InceptionV3 and Inception_Resnet_V2 models it has been reduced to 299x299. On the other hand, we must apply the same normalization that is applied to the images of the Imagenet dataset. Which consist of rescaled the pixel intensities between 0 to 1 and obtain zero mean and unit variance in terms of image data.

- Output Layer. We have formulated the problem as a multiclass classification task, so after the last layer of the convolutional model we must add a softmax layer. The softmax layer will assign a decimal probability to each class according to its relevance and the sum of the probabilities of all classes will sum 1. These characteristics provided by the softmax function allow a faster convergence in this type of problems and provides a valid probability distribution with values between 0 and 1 for each class. This output will be used as the input for stage 2.

Loss function: The loss function used is the categorical cross entropy (CCE) loss. Unlike the most commonly used loss function (the binary cross entropy) with CCE there is no limit to the number of classes that the model can classify. In our case we have used 3 classes.

Ensembling multiple predictions.
The prediction of each of the trained models will not be limited to a single prediction with the test dataset. Instead, we have implemented an algorithm, which in the schemes we call "Function_TTA", based on the Test-Time Augmentation (TTA) technique, whose operation and benefits have been explained in section 3.4. With this technique we obtain 5 predictions for each of the trained models (Densenet, Inceptionv4 or Inception-resnetv2). The following steps will be carried out for this purpose:

1. The requested model is built with the previously trained weights.
2. A random transformation is applied to the test set of scale, rotation, displacement and/or orientation.
3. The model calculates a series of predictions for the transformed test set.
4. The new prediction is stored in a list.
5.1. Stage 1: Multiclass-classification problem

With all the predictions obtained, an unweighted average will be applied to all the predictions of the same sample.

Ensembling multiple models.

Once each model that integrates the architecture gets an accurate prediction. Predictions from different models are combined in a weighted way. To select the weights applied to each prediction we will use a grid search algorithm. Where for each of the possible weight combinations we calculate the AUC-ROC scores for each class and the mean value, checking every possible combination. The combination selected was the one that maximized the average AUC-ROC score.

Figure 5.2 shows the architecture of the resulting classification stage 1 including the weights that maximize the objective metric.

5.1.2 Experiments & Results

In this section we will show the results obtained in both the training and prediction process applying the architecture and techniques described above.

Single model Training Methodology. Each convolutional model has been individually trained using the Imagenet’s pretrained weights. Each of them share the same techniques and hyperparameters used that are summarized in Figure 5.4.

In the optimization section, each layer has been trained by Adam optimizer with the standard configuration parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) using mini-batches of size 64 and going through the entire dataset in each epoch which means 320 steps.
Chapter 5. Classification module

Figure 5.5: Training process results in stage 1 of the classification module.

per epoch. Finally, starting from a learning rate of 0.001 this will be reduced on demand by applying a dynamic learning rate scheduler called "ReduceLROnPlateau". ReduceLROnPlateau defines a callback by which we monitor the value of validation loss, reducing hyperparameter learning rate by a factor of 0.5 after 2 epochs where the monitor does not detect improvement.

In the regularization section, in order to avoid the effects of overlearning (overfitting) we have applied: data augmentation and early stopping.

Through data augmentation we generate new samples applying light transformations to the original images. For this we have used the imgaug\textsuperscript{2} library with MIT License including the random transformations of: Horizontal Flip, Scale Variation, Rotation, Translation and changes in Brightness and Contrast. Finally, Early stopping allows us to stop the training when it detects that a metric, in our case, validation loss that does not improve after 6 epochs.

\textsuperscript{2}https://github.com/aleju/imgaug
5.1. Stage 1: Multiclass-classification problem

The graphs resulting from the training are shown in Figure 5.5. It is observed how Inception-Resnetv2, the most current model and at the same time the most computational demanding, shows better results in the training phase in terms of loss and accuracy.

**Ensembling predictions for each model.**

Once we obtain the trained models we apply the Test-time augmentation strategy explained in the previous section. Figure 5.6 shows the results, where three important aspects can be observed:

- All models show similar behaviour when detecting different classes. Obtaining better results in the detection of normal samples and worse results in the detection of non-normal samples.
- The model with the best results is InceptionResnetV2.
- It is surprising that the densenet121 model obtains better results than a more current model such as InceptionV3. In any case, the difference is small and not very significant.

**Ensembling models and final results obtained.**

The predictions of each model that we have evaluated previously have been combined in a weighted way. The weights that maximize the average AUC-ROC score and the results obtained with these weights are shown in Figure 5.7.

The results obtained reflect a significant improvement after combining the predictions of multiple models against the results of the individual predictions of each model (Figure 5.6). Figure 5.8 shows the resulting ROC curve for each of the classes.
Chapter 5. Classification module

Figure 5.8: ROC Curve of the final predictions in stage 1 of the classification module.

Figure 5.9: Initial Scheme of Stage 2 of the classification module.

5.2 Stage 2: Binary classification problem

In the second stage of the classification module we reformulate the problem as a binary classification task where in the output we will have a single binary output $y \in \{0,1\}$ that will represent the presence or absence of pneumonia in the sample.

For this purpose, an end-to-end network has been built based on a feedforward structure designed and trained to detect presence of pneumonia outputs $\hat{y} = P(y = 1 \mid x)$ estimating the probability that a sample contains signs of pneumonia from the input $x$ characteristics consisting by only 6 features: 3 extracted from the DICOM file (sex, age, view) and 3 corresponding to the distribution of probabilities of the normal, pneumonia, and non-normal classes from stage 1 of the classification module.

In Figure 5.9 we show the starting scheme of the architecture implemented in this second stage of the classification module.
5.2.1 Architecture design

For this task we have built a FFNN (Feedforward Neural Network) topology from scratch. The FFNN structure is one of the best-known base structures of the deep neural network theory that has already been introduced in this paper in subsection 2.2.1. In order to build a neural network from scratch we must fully define all the layers that make up the neural network as well as various aspects such as the initialization of the weights of all nodes, the activation function applied to each node and the implementation of various regularization techniques within the architecture.

The input layer.
The input layer is strongly correlated to the number and type of features in the input data. As the input vector X is composed of only 6 features, we have arranged on the input layer 6 nodes. These nodes implement rectified linear activation unit as an activation function and their weights will be initialized by the He initialization technique, a technique that presents the best results for layers with ReLu activation [11].

The output layer.
The output layer depends on the approach of the problem and depends on the defined cost function. Since this is a binary classification problem, we have placed a single node at the end of the neural network in which to use a sigmoid activation function. The sigmoid activation function is the activation function that works best with binary classification tasks in conjunction with binary cross-entropy loss [39].

The hidden layers.
Following the indication of the book [39] we will use again a ReLU activation function in the hidden layers together with the He initialization technique. However, in order to determine the number of hidden layers and the number of nodes per layer, there is no guiding theoretical principle in the related literature that allows us to determine exactly what is the best possible composition and arrangement. It will depend a lot on the type of problem and the input data features. Despite this, several studies have analyzed the influence of the number of layers on feedforward neural networks, creating some general recommendations:

- Although single-layer networks comply with the "universal approximation theorem" which states that a single-layer neural network can approximate any measurable function [78], the fact is that, in some problems the use of one layer is very inefficient compared to topologies that integrate more hidden layers. The study [79] proposes the use of at least two hidden layers.

- Lippmann emphasizes that the number of nodes must be high enough to form a decision as complex as the problem requires, but not so dense that it ends up memorizing the dataset losing generalization. This author proposes the use of 3
Chapter 5. Classification module

hidden layers when building a neural network [80].

In addition to defining the depth and width of the neural network, we must achieve with the neural network a correct generalization. As we have few input variables we could overfit the model very easily. For this reason, we have implemented the Dropout regularization technique in each layer.

In short, the variables you have to configure when building our neural network are:

- Select the optimal size (total number of nodes), depth (number of layers) and width (number of nodes per layer).
- Select the dropped-out probability of the Dropout regulation technique.

Since there is no analytical formula for calculating model performance with each of the possible structures, we used a trial and error method calculating the performance of a finite number of possible combinations. To do this, we have followed the following steps:

1. We establish what all possible combinations are:
   - Number of possible layers and nodes per layer used: [4,2], [8,4], [16,8], [32,16], [64,32], [8,4,2], [16,8,4], [32,16,8], [64,32,16] and [128,64,32].
   - Probability associated with the Dropout technique: [0:0.5:0.1].

2. We build the neural network given the number of layers, nodes per layer and the dropped-out probability.

3. We train this neural network using the train set and validation set samples.

4. We calculate the AUC-ROC score of presence/absence of pneumonia against the test set. If an improvement in the target metric is obtained, we record the combination applied.

5. Repeat steps 2 to 4 in a loop until all possible combinations are used.

The training methodology.

In the list of steps above, with each combination, the neural network is trained using the train and validation subset of the data.

In the optimization section, we have used the Adam optimizer with the standard configuration parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) and mini-batches of size 128. In addition, we will start the hyperparameter with a value of 0.001 that will be reduced through the functionality integrated in the library "ReduceLROnPlateau" that monitoring the variable "validation_loss" reducing in a factor of 0.5 after 3 epochs without improvement.
5.2. Stage 2: Binary classification problem

of this metric until reaching a minimum of 1e-9.
In the section of regularization, we will use the techniques Dropout and Early Stopping, this last technique will stop the training after 8 epochs without improving the variable validation loss.
Since the number of input features is very small and the topology is not very deep either, the average training time of each model is small, approximately 10 minutes. This makes it possible to train a multitude of models in the search for the best possible combination.

Characteristics of the topology with better performances.
In Figure 5.10 the configuration of the neural network is shown with better performances, which characteristics are:

- Number of layers: 3
- Number of nodes per layer: [16, 8, 4]
- Dropped-out probability of nodes: 0.35

5.2.2 Experiments & Results

Figure 5.11 shows the training process for the best performing structure. This structure allows us to maximize the ROC-AUC score in the binary classification of presence and absence of pneumonia in the whole radiography. Finally in Figure 5.12 we show the ROC curve, with this architecture in 2 phases we reach the ROC-AUC score of 0.8919 in the binary classification of pneumonia cases.
Comparing this value with the result obtained in the first stage of the classification model,
Chapter 5. Classification module

Figure 5.11: Results of the training process of the models of stage 2 of the classification module.

Figure 5.12: ROC Curve and AUC-ROC score of the output predictions from stage 2 of the classification module.

where in the presence of pneumonia a value of 0.8905 was obtained. It can be concluded that stage 2 of the classification module that integrates patient annotations provides a slight improvement in the classification task.
In this chapter we will describe the third and final module in which the proposed implementation is subdivided. As we saw in the previous chapter, the classification module performs a global classification of the whole image. On the other hand, this module (focused on object detection tasks) aims to delimit and classify locally zones inside the radiographic image in which signs or features indicating the presence of pneumonia. Providing for each of these signs the coordinates of the bounding box that delimits it inside the image and the confidence score that provides the probability that the anchor box contains that object.

The detection of objects has been a field within computer vision that has not had so much production of studies especially in the field of medical imaging compared to other tasks such as classification or segmentation.

The main reason for this low production is due to the need for a sufficiently large dataset with precise annotations of the bounding boxes that should be predicted by the object detector system.

At that point, the creation and dissemination by RSNA of a dataset focused on object detection in a field as specific as the detection of pneumonia features in radiographic images provides an enormous impulse in the development of specific CAD systems for that task. Allowing to explore and adapt the frameworks of detection of objects of generic use to more specific and complex tasks and cases such as the medical imaging.

In subsection 2.2.9 of this document, we conclude that the framework that is currently providing the best performance in terms of efficiency, speed and even accuracy is the framework of a step called Retinanet [4]. For this reason, throughout this chapter we will work with the Retinanet detector.

We have structured this chapter in two parts. In the first part, we will introduce the components that make up the Retinanet detector. We will describe how we have implemented it and what changes have been done with respect to the base architecture. Following the guidelines described in section 3.4, we have built a more complex architecture that combines the predictions of 2 trained detectors in a weighted way and we have also used the "Test-time augmentation" technique to improve the prediction of each detector.
In a second part, we will dedicate it to show the results of the experiments carried out. We will analyze the performance obtained with the changes applied to the Retinanet base architecture and we will select 2 backbones and input sizes that we will use in our created object detection architecture. Finally, we will expose the performance obtained by our completely built and trained architecture.

6.1 Composition of the Retinanet architecture

The object detection architecture called Retinanet was created by the Facebook AI Research (FAIR) team in 2018. In the paper [4] its creators detail which components constitute this detector, formulate the new loss function called "focal loss" and show the results obtaining the best results in accuracy, efficiency and speed among all the current object detectors. A detailed description of the Retinanet detector exceeds the project object, which focuses on its implementation and use. But now, we’ll introduce the main components that constitute the Retinanet architecture, allowing us to contextualize the changes we have made over the default architecture and that we going to explain later. Figure 6.1 shows the architecture of the Retinanet detector and its main components:

- **Backbone.** This is a convolutional neural network like those used in the classification tasks (in the paper use two backbones Resnet-50 and Resnet-101). This CNN will act on the image and extract the features maps, obtaining more and more complex representations as it passes through the different convolutional layers that it integrates.

- **Feature pyramid net (FPN) [81].** A feature pyramid network (FPN) is integrated into the Retinanet backbone. FPN will allow to extract at different levels multiple features maps of the backbone with different scales. Each of the levels then participates in the object detection task.

- **Anchor Boxes.** On each of the features maps of the previous levels, there will be multiple anchor boxes whose dimensions and aspect ratio have been previously
6.2 Architecture design of the object detection module

Once the basic composition of a Retinanet architecture has been described, we will go on to detail the implemented architecture and the changes and additions applied to it. As we did in the classification module, the architecture of the object detection module is not limited to a single trained Retinanet architecture, but following the guidelines described in section 3.4 will incorporate the technique of TTA and bagging combination, i.e., for each detector will calculate several predictions and then these predictions will be combined in a weighted way to obtain the output of the object detection module. Figure 6.2 shows the implemented architecture to develop the object detection module.

6.2.1 Default retinanet architecture implementation

The integrated Retinanet detector derived from Fyzir GitHub project with Apache License 2.0. This implementation faithfully reproduces the architecture proposed by the creators of the Retinanet Detector and at the same time allows us to configure the following parameters by command line:

- **Backbone.** As the original architecture we will use the Resnet architecture. In the paper use the networks Resnet-50 and Resnet-101. We include the network Resnet-152 incorporating a deeper CNN.

- **Input size.** The original size of the image is 1024x1024, but using its original size is totally unfeasible in terms of necessary resources and training time. Instead, we will train most of the configurations and techniques applied with an input image size of 256x256. And later, we explored different input sizes. Training detector for...
Chapter 6. Object Detection Module

Figure 6.3: List of backones, input size and batch-size used.

<table>
<thead>
<tr>
<th>BACKBONE NETWORK</th>
<th>INPUT IMAGE SIZE</th>
<th>BATCH SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESNET-50</td>
<td>256 x 256</td>
<td>32</td>
</tr>
<tr>
<td>RESNET-101</td>
<td>256 x 256</td>
<td>32</td>
</tr>
<tr>
<td>RESNET-152</td>
<td>256 x 256</td>
<td>32</td>
</tr>
<tr>
<td>RESNET-50</td>
<td>512 x 512</td>
<td>10</td>
</tr>
<tr>
<td>RESNET-101</td>
<td>512 x 512</td>
<td>8</td>
</tr>
<tr>
<td>RESNET-152</td>
<td>512 x 512</td>
<td>6</td>
</tr>
<tr>
<td>RESNET-50</td>
<td>768 x 768</td>
<td>5</td>
</tr>
<tr>
<td>RESNET-101</td>
<td>768 x 768</td>
<td>3</td>
</tr>
<tr>
<td>RESNET-152</td>
<td>768 x 768</td>
<td>2</td>
</tr>
</tbody>
</table>

input images of: 256x256, 512x512 and 768x768.

- **Batch-size.** The batch-size used will be determined by the memory limitations of the GPU used, using the highest batch-size allowed by our hardware. In Figure 6.3 we group the network and input size backbones used which conditions the allowed batch-size.

- **Config file - anchor boxes.** We may vary the list of anchor boxes used in retinanet’s architecture. It will be explained in more detail in subsection 6.2.2.

- **Image augmentation.** The paper facilitates the regularization by implementing a very limited number of transformations. As an improvement, a more marked data augmentation has been implemented, which will be detailed in section 6.2.3.

### 6.2.2 Anchor Boxes Density optimization

In a one-stage detection architecture selecting a correct anchor boxes density is key to cover all possible sizes and aspect ratios of bounding boxes that best fit and adapt to the problem and the dataset used.

In the original paper, Retinanet uses 9 unique anchors boxes: 3 sizes \{2^0, 2^{1/3}, 2^{2/3}\} and 3 aspect ratios \{1:2, 1:1, 2:1\}. These anchors boxes are distributed and moved on feature maps of each of the feature pyramid network (FPN) levels.

If we graph the distribution of the aspect ratios of the bounding boxes of the annotation files, Figure 6.4, we can see that the aspect-ratio 1:2 used in the original implementation would have a low influence on our dataset. Therefore, we have implemented a method to calculate the aspect ratios of the anchor boxes that best fit our dataset. In order to determine which choice of aspect-ratio will have greater impact and adapts better to our dataset, we can consider it as a clustering problem.

For this, we have implemented a variant of kmeans called Ckmeans.1d.dp [82] which is a
6.2. Architecture design of the object detection module

Figure 6.4: Distribution of the aspect ratio of the annotations in the bounding boxes.

Figure 6.5: Results of the optimization and increase in density of Anchor boxes.

specialized dynamic algorithm optimized for clustering problems with only one dimension. The original implementation is available in R, but we used a python import\footnote{https://github.com/llimllib/ckmeans}. We have grouped the aspect ratios of all bounding boxes that contain the files of annotations around 5 clusters, we obtain their centroids: 0.72 1.04 1.42 1.42 1.76 2.44. With this, we obtained up to 15 unique anchor boxes that we have used in each level of FPN. In Figure 6.5 we show the increase in anchor box density, where a clear increase in profiled anchor boxes can be seen and how these are better adapted to the bounding boxes searched for.

6.2.3 Regularization Techniques

Data augmentation.
The original Retinanet detector applies only a simple horizontal transformation image flipping as data augmentation. Because of the size of our dataset is significantly smaller than that used by the author in his tests and in order to obtain a greater generalization
and improvement in the training process, we applied a more intensive data augmentation. The transformations applied for each subset sample of training have been: Horizontal flip, change of scale and rotation, small displacements of the radiography on the image and finally small variations of the levels of brightness and contrast.

**Dropblock.**

In subsection 2.2.6, we have analyzed the convenience and benefits of using Dropblock [45] on convolutional neural networks. For its implementation we have used the python library "keras-drop-block" integrating the dropblock regularization on each one of the levels of the Feature pyramid net of the Retinanet default architecture. The Dropblock application is configurable through two parameters:

- **Block-size.** Defines the size of the deleted block each time the regularization acts.
- **Keep-probability.** Sets the frequency with which it is applied to the regularization.

In subsection 6.2.1, we show the results obtained with different Dropblock configurations, analyzing the impact on detector performance. Being the Block-size = 4 and keep-probability=0.9 the configuration that best adapts to our project.

### 6.2.4 Score and NMS Filter of the output predictions

Once we have the trained object detectors, they provide us with a large number of predicted bounding boxes for each input sample that needs to be properly filtered:

- Bounding boxes with a low confidence score, which implies a low probability of framing the desired objective, so it cannot be considered a correct prediction.
- Several bounding boxes that successfully frame the same object can be detected for the same detection. It is necessary to filter, deciding which of the predicted bounding boxes best approximates the bounding box searched for.

In order to filter the predictions, a filtering algorithm will be applied to the output of each model, where the following steps take place sequentially:

- **Score Filter.** All bounding boxes with a confidence score below a preset confidence threshold (Th\textsubscript{score}) are discarded.
- **NMS Filter.** A list is created with all bounding boxes sorted by score confidence. For each element in the list, we calculate the value of IOU (Intersection over Union) with the rest of the elements in the list. If the result of this calculation exceeds a given threshold (Th\textsubscript{nms}) we consider that both bounding boxes are pointing to the
6.2. Architecture design of the object detection module

Figure 6.6: Representation of radiographic predictions before and after filtering.

same object, so the bounding box with the lowest confidence score is removed from the list. Once the whole list has been scrolled, the result will be a list of bounding boxes not duplicated with the highest confidence score for each detected object.

In Figure 6.6 we show an example of the application of the algorithm. The image on the left shows all the bounding boxes detected. And the image on the right shows the result of the cleaning carried out by the filtering algorithms.

To define the optimal thresholds we have used again a grid search mechanism where we will try with all the combinations of decision thresholds "Th\textsubscript{score}" and "Th\textsubscript{nms}" possible obtaining which ones maximize the objective metric that in our case is AP\textsubscript{@[.4:.75]} or AP\textsubscript{RSNA} 3.5.2.

6.2.5 Ensembling methods

As in stage 1 of the classification model, we will use the Test-time augmentation technique, combining for each model a total of 5 predictions for different versions of the test subset transformed by data augmentation technique. The same weight was applied to each prediction.

And on the other hand, we will combine the predictions of 2 models in a weighted way where the weights assigned to each model are obtained again by the grid search mechanism, discovering the combination of weights for the predictions of each model that maximize the objective metric, in our case, AP\textsubscript{rsna} Section 3.5.2.

Although this strategy is similar with respect to the classification module, its implementation involves certain complications. Such complications lie in the format and nature of object detector predictions:
Chapter 6. Object Detection Module

- The prediction of each of the detectors follows the following format \([x_1, y_1, x_2, y_2, \text{confidence score}]\).
- The detectors for the same image will be able to offer one or several bounding boxes framing according to the quantity of entities have been able to detect.

Therefore, when we consider the algorithm that combines these predictions we will find ourselves with the situation: Given a single input sample image, each of the "M" models will generate a "PM" prediction composed of multiple bounding boxes "BN". Ideally for each image all models will generate the same number of bounding boxes and all detectors will detect the same objects in the image. This ideal situation will not occur most of the time, so the assembly algorithm must take into account the following premises:

- Two prediction boxes will be considered correlated and pointed to the same object, if the IoU (intersection over the union) calculation of two boxes exceeds a predefined threshold of 0.5 (50%).
- A bounding box that is not found in all the predictions to be combined will have a penalty in its final confidence score depending on the number of absent predictions.
- Since confidence scores can be penalized, the algorithm must integrate an output prediction filter based on score filter threshold \((\text{Th}_{\text{score}_\text{ens}}))\).

Under these premises we have integrated an open source implementation available in Github\(^2\) to perform the assembly.

6.3 Experiments & Results

Training methodology.
In the optimization section, we use again the Adam optimizer with the standard configuration parameters \((\beta_1 = 0.9\) and \(\beta_2 = 0.999))\). The size of mini-batches depend on the backbone and input size used according to Figure 6.3. We start the training with a learning rate of 0.0001 that will be reduced depending on the monitoring of the variable "validation-loss", reducing by a factor of 0.2 after 2 epochs without improvement of this metric until reaching a minimum of 1e-9.

In the regularization section, we will use the Early Stopping technique to stop the training after 6 epochs without improving the variable validation loss and we will explore the implementation of Dropblock with its most optimal parametric configuration.

Evaluation of changes in density of anchor boxes and Dropblock.
First, we will evaluate the improvement introduced by the changes applied to the Retinanet

\(^2\) https://github.com/ahrnbom/ensemble-objdet

58
6.3. Experiments & Results

Figure 6.7: Results obtained with the anchor boxes density optimization and different Dropblock configurations.

<table>
<thead>
<tr>
<th>BACKBONE NETWORK</th>
<th>INPUT SIZE</th>
<th>AP_{RSNA}</th>
<th>TIME / SAMPLE (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESNET50</td>
<td>256</td>
<td>22,198</td>
<td>132</td>
</tr>
<tr>
<td>RESNET101</td>
<td>256</td>
<td>22,428</td>
<td>129</td>
</tr>
<tr>
<td>RESNET152</td>
<td>256</td>
<td>22,467</td>
<td>135</td>
</tr>
<tr>
<td>RESNET50</td>
<td>512</td>
<td>22,606</td>
<td>163</td>
</tr>
<tr>
<td>RESNET101</td>
<td>512</td>
<td>22,830</td>
<td>167</td>
</tr>
<tr>
<td>RESNET152</td>
<td>512</td>
<td>22,719</td>
<td>122</td>
</tr>
<tr>
<td>RESNET50</td>
<td>768</td>
<td>22,517</td>
<td>240</td>
</tr>
<tr>
<td>RESNET101</td>
<td>768</td>
<td>22,319</td>
<td>297</td>
</tr>
<tr>
<td>RESNET152</td>
<td>768</td>
<td>21,304</td>
<td>370</td>
</tr>
</tbody>
</table>

Figure 6.8: Results obtained with the different backbone networks and input image size.

architecture of the Retinanet architecture, that is, an anchor boxes density optimization and regularization techniques. To do this we will train the shallower backbone, i.e. Resnet-50 and the input image size resized to 256x256 pixels.

The results of these experiments are shown in Figure 6.7 We can see that training the Retinanet detector with the anchor boxes density optimization for our dataset we get better performance.

And using this density of anchor boxes we evaluate the performance obtained after applying the Dropblock control technique with different configurations of block-size. The configuration with the best results is to apply the optimization of anchor boxes together with Dropblock with keep-prob=0.9 and block-size=3.

**Evaluation of different backbone and input image sizes.**

Secondly, we will evaluate the performance obtained with different backbone and input image size.

In each of the detectors we have trained we have used the optimized configuration of anchor boxes and Dropblock. The results have been grouped in Figure 6.8.

The two models that offer us the best performance in terms of AP_{RSNA} metrics are the detectors that combine the backbones Resnet-101 and Resnet151 on a 512x512 input image. It should be noted that although with an image size of 768 the computational time and resources required increases considerably, no improvement can be seen. The
reason for this is that the hardware limitations used force us to reduce considerably the batch-size used, which affects the training phase and the performance of the detector obtained.

**Final Output. Ensembling predictions.**

Once trained and selected the two detectors that we will incorporate in our implementation. As we can see in the module architecture Figure 6.2 we combine the predictions of each detector. In Figure 6.9 we group the performance of each of the 5 predictions obtained by applying the Test-time augmentation (Function_Prediction_TTA) technique and going through the Score and NMS filters. These filters are configured with the optimal thresholds (Th\text{score} and Th\text{nms}) that are also shown in the table.

Each of the 5 predictions are combined assigning the same weight to all the predictions. The table shows the AP\text{RSNA} value that obtains the prediction gotten by each detector. The predictions of the two detectors are combined in a weighted way. The optimal score threshold (Th\text{score}_{ens}) and values of the weights are shown in the table as well as the final result obtained by the entire object detection module.
7 Complete final architecture

Between chapters 4 and 6 we have gone through all the modules that compose the constructed diagnosis architecture, describing the design aspects and the most important actions that take place. In this itinerary we have also progressively evaluated each of the modules separately.

In this chapter we unify all the modules showing the complete implementation, doing a concise review of each module. As a new addition, we explain the connection we establish between the output of the classification module and the predictions of the object detection module in order to improve the performance of the latter. And we will evaluate the possible improvement provided by this approach.

Finally, we will evaluate the final results obtained by the full implementation in context against similar works.

7.1 Architecture Design Summary

Figure 7.4 shows the complete architecture that combines the 3 modules explained in the previous chapters. Next, we will make a brief review of each of the modules that compose the complete architecture. In this review we will remember the most important elements and objectives of each module:

1. Preprocessing Module. This module is the first contact of the architecture with the input sample. In its interior it presents a DICOM data extractor that extracts the useful information from the input DICOM archive, obtaining the radiographic image and some annotations associated with the patient. It also has an algorithm that detects and eliminates black edges within the image, making the radiography centered and occupying as much space as possible in the image.

2. Classification Module. Specialized module to classify chest x-rays. It is subdivided into 2 stages:
(a) Stage 1. The radiographic image is classified in one of the 3 possible classes: normal, lung opacity and non-normal. It is integrated by 3 CNNs (Resnet-121, InceptionV3 and InceptionResnetV2). Each model records 5 predictions. These 5 predictions are combined by a simple average and finally the predictions of the 3 models are combined in a weighted way.

(b) Stage 2. The result of stage 1 is combined with annotations extracted from the DICOM file. An FFNN network provides the probability that the x-ray is a case of pneumonia or not. In other words, it is a binary classification problem.

3. Object Detection Module. Detects areas with opacities in the lung that may be a case of pneumonia. The result will be a list of bounding boxes with their corresponding score confidence with the areas that you have detected with signs of pneumonia in the chest x-ray. This module implements two Retinanet detectors with different backbone networks (Resnet-101 and Resnet-152). These detectors use the 512x512 image size and will calculate 5 predictions each. These predictions will go through scoring and nms filters previously adjusted to the values of \( T_h_{\text{score}} \) and \( T_h_{\text{nms}} \). The 5 predictions will be combined without weighting to obtain the prediction for each detector. Finally, these two predictions are combined by weighting the predictions and these predictions go through the final score filter with the threshold \( T_h_{\text{score\_ens}} \).

From this complex proposed architecture a very complete diagnostic information is obtained for the medical professional:

- Global information. Binary classification of the entire x-ray, indicating the likelihood that the x-ray will contain evidence of pneumonia.
- Local information. In cases where the x-ray shows areas with signs of pneumonia, these are framed within a bounding box and a score confidence.

7.2 Enrichment object detection with global classification information

With the architecture described above, we would have a system that acts as a detector and classifier but there is no interaction between the two modules, so they could work in parallel without interacting with each other. In this section we propose to go one step further and integrate the output of the classifier (with global classification information) into the object detector. Assuming that the modules work sequentially, we would already have a global classification of the radiography from the classification module before the object detector works.

Each of the detectors provides a list of predictions with the format \([x_1, y_1, x_2, y_2, \text{confident score}]\). We have applied a weighted average between the confidence score values of all
7.3 Experiments & Results

On this occasion, it was not necessary to train any models, we used the previously trained models. In this section we show the results obtained after implementing the integration between the classification module and the object detection module (Figure 7.1).

A slight improvement can be seen, going from a score of 24.801 (Section 6.3) to 24.960. For this, the output of the classification module is weighted by 8%.

As we did in the previous chapter, we grouped all the weights and thresholds that optimize the target metric together with the results presented.

The complete implementation with weights, thresholds and integration between the two modules is shown in Figure 7.4.

### 7.4 Evaluation in context

#### 7.4.1 Classification performance

As indicated in subsection 2.1.2, within the related works, the work done by the Stanford faculty called Chexnet stands out. This work is one of the reference works in the classification of cases of pneumonia in chest x-rays and in which only one convolutional

---

**Figure 7.1: Final results obtained in the task of detecting objects by the architecture built.**

<table>
<thead>
<tr>
<th>BACKBONE NETWORK</th>
<th>INPUT SIZE</th>
<th>THScore</th>
<th>THings</th>
<th>WClass</th>
<th>APRSNA</th>
<th>TTA ENSEMBLING</th>
<th>MODEL ENSEMBLING</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESNET101</td>
<td>512</td>
<td>0.4</td>
<td>0.45</td>
<td>0.08</td>
<td>23,729</td>
<td>23,324</td>
<td>24,960</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23,774</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23,751</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23,609</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESNET152</td>
<td>512</td>
<td>0.4</td>
<td>0.4</td>
<td>0.08</td>
<td>23,244</td>
<td>23,256</td>
<td>23,418</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23,244</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23,354</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
neuronal network is trained, more specifically a Densenet-121.
The dataset used in CheXNet’s work is the NIH dataset, which as we have indicated in section 3.2 our dataset is a subset of that dataset, but introduces certain improvements described in subsection 3.2.1. Because datasets have a common origin and the approach of the problem is similar. We will use CheXNet’s work to contextualize our results. Extracted from the paper [4] the AUC-ROC score reported by CheXNet in the binary classification of presence and absence of pneumonia is 0.833 and its ROC curve is shown in Figure 7.2. This reveals that our proposal significantly improves the data obtained in the Chexnet work. With an AUC-ROC score of 0.892, it shows the improvement in the task of classifying pneumonia cases in our implementation.

There are several reasons for this improvement:

- The use of a dataset that introduces significant improvements with respect to the original NIH dataset.
- The problem is presented in two phases and integrating the additional information provided by the DICOM file.
- Use of more recent classification models. Although we implemented the architecture of Densenet121, we have also integrated and trained two more current CNNs such as InceptionV3 and Inception-ResnetV2.
- The improvement introduced by the use of combination technique. Both the combination of multiple predictions for each trained model (TTA) and the combination of multiple models.

The above results illustrate the improvement in terms of the AUC-ROC score in the classification of chest x-rays provided by our implementation.
7.4. Evaluation in context

7.4.2 Object detection performance

In contrast to the classification task, there is currently no published work focused specifically on tafea for the detection of objects in pneumonia zones. But to contextualize the results obtained we will analyze the ranking of results of the contest where the dataset was born: "RSNA Pneumonia Detection Challenge". This contest took place between August and October 2018, hosting more than 1400 participants worldwide. After finalizing the contest a leaderboard has been elaborated with the best results (Figure 7.3).

The values shown in the Figure 7.3 use the metric proposed by RSNA in this championship which is the same one we have used throughout the project. Our proposal obtains a result of 24.96%, that is to say, 0.2496 being placed in the top2 of this classification. Highlighting the difficulty of achieving better results and the good results obtained.

Figure 7.3: RSNA Pneumonia Detection Challenge Leaderboard.
Chapter 7. Complete final architecture

Figure 7.4: Complete final architecture.


8 Conclusions and future work

8.1 Conclusions

From the beginning of this master’s thesis, our aim has been to develop an implementation that would allow us to accurately diagnose cases of pneumonia within a chest x-ray, providing a binary classification, of presence/absence of pneumonia, of the entire x-ray and identifying in it by bounding boxes boxes the areas with the presence of signs representative of pneumonia. For its realization we have used the recently created dataset "RSNA pneumonia detection challenge dataset", composed of multiple medical samples present in DICOM format and the necessary annotations with which to train and optimize each of the supervised models that compose our implementation.

Specifically, the implementation of this research project has been designed around 3 main modules, in order to organise its development and achieve the corresponding objectives more effectively. These are: preprocessing module, classification module and object detection module. This division is reflected in the structuring of this document in which we describe and evaluate each of these modules.

In the first place, using the preprocessing module we have extracted the information contained in the files in DICOM format towards a more manageable format, obtaining for each file a radiographic image and information associated with the patient. Subsequently, by employing an algorithm we have reduced the black frames and centered the radiography for each medical image. This algorithm has allowed to report an improvement in terms of clustering of the location information of the bounding boxes defined in the annotation files.

In second place, the classification module is proposed with a two-stage approach focused on the classification of radiological images. In the first stage, 3 different state-of-arts convolutional neural networks (Denset-101, InceptionV3 and Inception-Resnetv2) have been implemented, which together with the application of advanced techniques of regularization and combination of predictions, have allowed us to classify radiographic images in 3 classes: Normal, lung-opacity and non-normal, with an AUC-ROC score of 0.8905 for cases of pneumonia. In the second stage, we have created a neural network with
Chapter 8. Conclusions and future work

FFNN topology from scratch, whose structure has been entirely selected to maximize the quality of prediction in cases of binary classification (presence or absence of pneumonia); using as input the output of the classifier in 3 classes of the first stage and the information associated with the patient extracted from the DICOM file. In this second stage the predictions of presence or absence of pneumonia in the image are obtained with an AUC-ROC score of 0.8919. These results highlight the improvement resulting from the inclusion of additional patient data within the prediction of the medical image.

In the third place, the **object detection module**, specialized in the detection of areas that constitute signs of pneumonia. To do this, we have implemented the state-of-art Retinanet detector using as backbone a convolutional Resnet neural network. Subsequently, a series of improvements have been introduced to the architecture of the detector, such as the optimization of the anchor boxes used by the detector, the application of the Dropblock adjustment technique and, at the same time, the use of different sizes of image input and depth of the convolutional network used as a backbone. All these modifications have been evaluated obtaining the best results through the use of optimized anchor boxes, the use of dropblock and the resizing of the images to 512x512, as well as the use of the backbone resnet-101 and resnet-152. After combining these two detectors we have reached predictions with an AP@[.4:.75] score of 0.2480.

Finally, we have analyzed the combination of the output of the classifier module and the predictions calculated by each detector and, in this sense, the integration of the global information of radiograph classification -in the tasks of object detection- has allowed us to achieve a slight improvement in the quality of precision, obtaining an AP@[.4:.75] score of 0.2496.

The main objective as well as the specific objectives have been achieved satisfactorily throughout the course of this master’s thesis. If we contextualize the results obtained we can appreciate the good performances achieved despite the difficulty of the challenge we have faced:

- In the classification task. The reference work in the diagnosis of cases of pneumonia called CheXNet reports an AUROC score of 0.833. Our implementation provides an AUROC score of 0.892, which represents a significant improve in terms of the ability to distinguish between samples with presence and samples without the presence of pneumonia.

- In the object detection task. Our AP@[.4:.75] score of 0.2496 places our implementation in the top-2 into the leaderboard of the competition "RSNA pneumonia detection challenge". Competition that hosted more than 1200 participants worldwide around the same problem, the detection of areas on a chest x-ray with signs that constitute a case of pneumonia.

In addition, this research project provides a valuable and extensive overview of the performance and limitations of the dataset used, both in classification and object detection
8.2 Future work

As mentioned in the previous section, this master’s thesis covers the design and evaluation of a Deep Learning implementation, in the course of which a series of improvement aspects have been observed.

On the one hand, with regard to the dataset, we observe 3 evident aspects of improvement:

- To use the same identifier for X-rays of the same patient. Currently, due to privacy policies, each sample is identified with a different unique identifier even if they come from the same patient.
- To record for each patient at least one frontal X-ray and one lateral X-ray. Currently this dataset only has frontal captures, losing a very valuable information and with which professionals normally have to carry out their diagnoses.
- To have available the patient’s clinical information. Cases of pneumonia are usually preceded by a history of fever and cough. So, having access to this information - when it comes to making an accurate diagnosis - would give us a more accurate diagnosis.

On the other hand, we have used a variety of state-of-arts convolutional neural networks created for generic use, which we have adapted and trained for our project. In this regard, recently have appeared the first neural networks created integrally by reinforcement learning [53] techniques. It might be interesting to study the possibility of creating from scratch the convolutional neural networks used in diagnosis, optimizing them for our project and for this type of problems through reinforced learning techniques.

At the same time, in the development of the implementation we have sought to maximize the performance in terms of quality of predictions obtained through a complex architecture involving several neural networks. This implementation itself is computationally demanding, requiring high hardware resources for its execution. It might be possible to open a line of research that filters the models that provide the least added value or the use of models with a lower calculation requirement in order to obtain a more effective application at the level of computational resources required.

Finally, given the implementation developed and the results obtained, a CAD could be created that integrates this implementation, giving it an application format for use in professional environments and laboratory tests in real cases as support to the professional.
Bibliography


Bibliography


