

École polytechnique de Louvain

Building an Interpretable Deep Learning Model for Pneumology using Transfer Learning

Author: **Laureline WILLEM**
Supervisor: **Sébastien JODOGNE**
Readers: **Charles PECHEUR, Dany RIMEZ**
Academic year 2023–2024
Master [120] in Computer Science and Engineering

Acknowledgement

First of all, I would like to extend thanks for the many people that were involved in this masters thesis, starting with all members of the LIME (Health informatics lab), in particular Jodogne Sébastien, my supervisor, for his mentoring, Langlois Quentin, for his advises on AI technologies and redaction tips and Vanderhaeghen Juliette, for proof reading this manuscript.

Special thanks to the radiology service of the Clinique Saint-Pierre Ottignies, in particular Dr Arnould Marie-Cécile, radiologist, Dr Bosschaert Pierre, head of service, Quentin Conroux and Amine Dinia, radiology assistants for discussing this thesis and the modalities in the images acquisition at length as well as annotating the images, and Eloïse Dejardin, secretary and IT coordinator of the service, for welcoming me and helping with the technical aspect of the annotation process.

Special thanks also to Moïny Véronique and Morelle Isabelle for helping me to reach out to healthcare professionals, without whom the annotation collection would not have taken place.

I would like to extend my sincere thanks to my parents, Fabry Isabelle and Willem Alex, for their help in making my experiments possible and proof reading of this manuscript.

Lastly, I would like to mention my closest friends, for all their unconditional support during this year.

Contents

1	Introduction	1
2	Theoretical Background	3
2.1	Training, validation and testing sets	3
2.2	Convolutional Neural Networks (CNN)	3
2.2.1	Neural Network	4
2.2.2	Back to CNNs	5
2.3	Layers of interest	6
2.4	Transfer Learning	8
2.5	Grad-CAM	8
3	State-of-the-art	10
3.1	Deep Learning and Radiology	10
3.2	Pneumonia	11
3.2.1	NIH's chestX-Ray8 dataset	11
3.2.2	Chest X-Ray Images for Pneumonia Detection Dataset . . .	11
3.3	Objectives	12
4	Datasets	13
4.1	Labels	13
4.2	Data Exploration	14
4.2.1	Chest X-Ray Images for Pneumonia Detection Dataset . . .	14
4.2.2	NIH's chestX-Ray14 dataset	15
4.3	Annotations	19
4.3.1	Annotations Collection	19
4.3.2	Discussion and insights given by healthcare professionals . .	21
4.3.3	Results	22
5	Neural Network	24
5.1	Machine specification	24
5.2	Preprocessing	25

5.3	Data Augmentation	25
5.3.1	Image Random Transformation using tf.image	26
5.3.2	Noise	27
5.4	Architecture	28
5.4.1	Backbone	29
5.4.2	Complete Network	31
5.5	Training	31
5.5.1	Outer layer training	31
5.5.2	Unfreezing the backbone and fine-tuning	33
5.6	Knowledge transfer between datasets	33
5.6.1	Discussion and future work	35
6	Explainable AI (XAI)	36
6.1	Technology Choice	36
6.2	Methodology	36
6.3	Metric	37
6.3.1	Extraction of the Bounding Boxes	37
6.3.2	Computation of the IoU	38
6.4	Treatment of the annotations	39
6.5	Results	39
6.6	Heatmap Analysis	39
6.7	Discussion and Future Work	40
7	Conclusion	44

Chapter 1

Introduction

Medical imaging is one of the most prominent tools in modern medicine. This technology is used for diagnosis, for follow-up after recovery and sometimes even treatment. This leads to an immense number of images (the INAMI reports over 2.35 millions of Computed Tomography Scans and close to 1.16 millions Magnetic Resonance Images registered in 2019 in Belgium [1]). This volume of images has to be analysed by the radiologist before being submitted to the prescriber. This induces delays in diagnosis, since there is a delay to take appointment to get the image taken, and then the additional few days required for radiologist to analyse and upload said image. Deep learning thus started to peak the interest of the healthcare sector due to its utilization in computer vision [2] and has yielded to successful results [3]. This allows for better analysis of images and a more stable analysis, as the model will always use the same parameters for all inputted images, whereas clinician might be unconsciously biased by conditions.

Many hospitals cannot afford complex imaging devices and required staff to read and analyse the extracted data. Deep learning tools mentioned earlier are often costly and only available to state-of-the-art hospitals. Having such models available in Open Source could allow many hospitals to be more efficient and treat patients faster.

Pneumonia is a fairly common respiratory infection both for adults and children. According to the World Health Organisation (WHO), symptoms range from coughs to difficulties breathing. Some infected infant present fever, loss of consciousness and convulsion. In 2019, pneumonia killed 740 180 children, making the disease responsible of 14% of deaths of patient under the age of five years old [4].

Pneumonia Diagnosis is done through clinical analysis often accompanied by Chest X-Ray, enabling Deep Learning techniques such as image processing to assist the diagnosis.

This manuscript offers a deep diving in the creation of PneumonAI, an interpretable Open Source Convolution Neural Network (CNN) for pneumonia classification for children. PneumonAI will classify X-Ray images of children in two categories defining whether or not pneumonia has been detected and return its prediction and a heatmap highlighting features observed by the model to take its decision.

First, theoretical concepts necessary to understand the implementation will be explained to ensure accessibility of the rest of the manuscript.

Next, current advances in the literature will be analysed in order to establish the state-of-the-art in the domain of infantile pneumonia diagnosis using CNN.

As the background is fully set, the rest of the document will follow the path of the X-Ray data, starting by introducing and exploring the datasets used. In the same chapter, annotations were collected and healthcare professional offered insights on the dataset and the way radiology diagnoses pneumonia.

Once the data is ready, it will cover the CNN that was built. The next chapter will describe the preprocessing of the data, application of data augmentation and the architecture of the model as well as its performances. It also describes an experiment attempting to use knowledge gained on children X-rays to learn to classify adult X-rays.

To end the data pipeline, the penultimate chapter will discuss implementation of Grad-CAM, a tool enabling CNN interpretability, and comparing the features of interest extracted by the tool with the ground truth offered by the annotations collected.

Finally, the manuscript will conclude by an overview of the results obtained, the assessment of the success of the various objectives set for this master's thesis and suggestions for future work based on the results obtained by the end of this year of work.

Chapter 2

Theoretical Background

The following section introduces theoretical concepts necessary to develop PneumonAI, starting with training, validation and testing sets. Their proper definition and separation will ensure accurate evaluation of the resulting model. Then, the focus will advance to Neural Networks and Convolutional Neural Networks (CNNs), as it is the main tool used to cast predictions. To further understand the specifics of the architecture, all layer types used in the network are described. Once the network is build, as it will be exposed, it will need to be trained by gradient descent. A method allowing faster training called Transfer Learning has been used and is also covered by this section. Finally, the inner workings of Grad-CAM, the explainable AI (XAI) tool used to induce interpretability is exposed.

2.1 Training, validation and testing sets

Dataset are usually separated between three categories: training, validation and testing sets. Each set has its own task : the training set is used to train the model, the validation set to tune its hyper parameters, as it will be discussed in Section 2.2.1, and the testing set is only used to compute evaluation metrics. Each dataset should have a representative selection of the data, without any overlapping between sets. This ensures realistic evaluation, as models evaluated on training data tend to be overly simplistic[5].

2.2 Convolutional Neural Networks (CNN)

A Convolutional Neural Networks (CNNs) is a type of Deep Learning Network often used for tasks related to computer vision, as it is designed to interact with images. CNNs are a special instance of Neural Networks (NNs), which is why this parent concept should be exposed before dwelling in their convolutional variant.

2.2.1 Neural Network

Neural Networks, as the name suggests, are inspired by the human nervous system. They are composed of various nodes (the neurons) that interact with each other. They are organised in subsequent layers, which only communicate together in order. Neurons from layer n only take input from neurons from layer $n - 1$, and send their output to neurons of the layer $n + 1$. The only exceptions are the first and last layer, designed to accept the input and the output, respectively. All the other layers are called hidden layers, as they do not interact with data directly, which is transformed in the input layer. Each neuron applies a mathematical operation on its input, the activation function, and then outputs its result to the neuron(s) that use it as input in the next layer. Each link between neurons has a weight. By training the network, we adjust those weights by minimizing an objective function: the loss [6].

The training is a repetition of two phases, the forward propagation and the backward propagation. During the first (forward) step, the training set is processed normally, with the weights that are available at the moment: initial weights if it is the first cycle, or the weights resulting from the previous cycle. Thanks to the output from this first phase, the second (backward) step can compute a gradient of the objective function (the loss) and adjust the weights by gradient descent. This is applied through all layers starting from output to input, hence the name. By repeating this cycle until convergence, the model should get better at the task fixed by the loss function [6].

Hyper Parameters

The training set is used to train, and the testing set is kept aside to avoid influencing the result. One could wonder what purpose serves the validation set. It is used to adapt hyper parameters. These parameters are weights that infer condition on the learning process of the network instead of its structure. In keras, it is transmitted between each cycle in order to allow the programmer to use it. In this project, it was mostly used to reduce the Learning Rate during fine tuning. The learning rate is a hyper parameter that fixes how much the weights should be modified during back propagation. If it is too high, it might pass the optimal value when encountered, by making it too small, it will cause the process to be prone to converge on local extrema instead of finding the global one [7].

Gradient Descent

The gradient descent is an optimization algorithm. This master's thesis uses one of its variation called Stochastic Gradient Descent (SGD). On each iteration and

for each training sample, parameters are updated as such:

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta, x, y)$$

where θ are the parameters, η is the learning rate, x is the sample, y is the label, J is the objective function (the loss) and ∇ is the gradient operator [8].

Adaptive Moment Estimation (Adam) Optimization follows this process by computing per parameter adaptive learning rate following the formula:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

where ϵ is a parameter, suggested to be set to 10^{-8} [9] and $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moment estimate.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1}$$

where β_1 is a parameter, suggested to be set to 0.9 [9] and m_t is the first moment estimate.

$$\hat{v}_t = \frac{v_t}{1 - \beta_2}$$

where β_2 is a parameter, suggested to be set to 0.999 [9] and v_t is the second moment estimate [8].

Loss

Binary Cross Entropy (BCE) loss is a common loss function choice for binary classification on images [10]. The mathematical formulation is given as follows:

$$BCE(y_i^t, y_i^p) = - \sum_{i \in S} y_i^t \cdot \log(y_i^p) + (1 - y_i^t) \cdot \log(1 - y_i^p)$$

where S is the set of sample, y_i^t is the true label for the i^{th} sample and y_i^p is the predicted label for the i^{th} sample.

2.2.2 Back to CNNs

In order to treat images, CNNs use a variety of components, that will be described hereafter for a better understanding of the functioning of such networks. This section is based on "*CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope*" [11].

The first particularity is obviously that the input is in the form of an array, representing the image. Indeed, the image is represented by a tensor containing one array per canal, and each array a two dimensions mapping of pixel values, ranged between 0 to 255, as it is written on 8 bits. Gray scale images only use one canal [11].

What has given this network its name is the use of the 2D convolution operation. Some layers, called convolution layers, apply the operation on the input image with small 3×3 pixels or 5×5 pixels filters. The formula is a double sum (as the image is a discrete space) of a matrix multiplication:

$$o[x, y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m, n] \cdot i[x - m, y - n]$$

where o is the output, i the input, f the filter, x & y the coordinates and m & n are dummy variable used stride in the full domain [12].

By choosing the right filters, the resulting output is either smaller or the same size as the input. The goal is to enhance and extract features from the image, which allows the following layers to use the result to solve the task. First layers extract fairly simple data from the image, such as vertical and horizontal characteristics enabling further layers to extract increasingly complex features [11].

In order to reduce computing power needed, pooling layers are placed after convolution layers in order to reduce their dimensionality. This is usually done through 2×2 filters that pool maximum or average value from the analysed region [11].

Finally, the resulting features are used to solve the task before outputting the result [11]. In this case, the features are used to classify the image in one of our two categories: PNEUMONIA or NORMAL.

2.3 Layers of interest

A plethora of layer types have been implemented in keras, each doing its own transformation. The following list describes the usage of all layers used to implement the CNN in Chapter 5:

- **Global Average Pooling 2D layer:** a pooling layer based on average used to downsample the features extracted by the backbone specifically designed

for 2D images.

$$GAP2D := f(x) = \frac{1}{Z} \sum_{i=0}^n \sum_{j=0}^m x_{ij}$$

where n is the height of the image, m is the width of the image, $Z = n * m$ is the total number of pixels and x is the input. Reducing the dimensionality should minimise risks of overfitting [13];

- **Dropout layer:** deletes feature detectors following a fixed probability.

$$Dropout := f(x) = \begin{cases} 0 & \text{for } \frac{1}{rate} \text{ elements} \\ x * \frac{1}{1-rate} & \text{otherwise.} \end{cases}$$

It prevents co-adaptation (using the same feature multiple times during the extraction) and has an action against overfitting as well [14];

- **Dense layer:** dense layers are connected to all the neurons of the previous layer. The formula is given by the keras documentation [15]:

$$Dense := f(x) = (x \cdot kernel) + bias$$

where the kernel is a weight matrix and bias is a weight vector, both related to the layer.

- **Rectified Linear Unit (ReLU) activation function:** makes sure outputs are positive or replace them by zero. It is frequently used in CNNs [11]. Mathematically, it can be defined as follows:

$$ReLU := f(x) = \max(0, x)$$

MobileNet uses a particular version, called ReLU6 that also caps value to six, make the full formula:

$$ReLU := f(x) = \min(\max(0, x), 6)$$

- **Batch Normalization layer:** as the name suggests, normalises its input. It keeps the mean of the output close to zero and its standard deviation equal to one. The formula is given by the keras documentation [15]:

$$BatchNormalization := \gamma * \frac{batch - moving\ mean}{\sqrt{moving\ var + \epsilon}} + \beta$$

with ϵ a parameter fixed at declaration to avoid division by zero, and γ and β two hyper parameters. The moving mean and variance defined as follows, and updated on each training epoch:

$$moving\ mean = moving\ mean * momentum + mean(batch) * (1 - momentum)$$

$$moving\ var = moving\ var * momentum + var(batch) * (1 - momentum)$$

Emphasis should be put on the fact that the moving mean and variance are only modified during training, and kept as is during inference.

- **Depthwise 2D Convolution:** form of 2D convolution that keeps channels separated.
- **Zero Padding 2D layer:** pads borders of the image received as input.
- **Add layer:** performs basic elementwise addition operation on its inputs.

2.4 Transfer Learning

Building a deep learning network from scratch is an extensive task, notorious networks contain millions of parameters, which can take days to train. The time invested usually allows for better results but can makes it difficult for people to start developing tools using the technology. From that need, emerged an idea: what if some models are general enough that one could reuse them for various tasks with minimal training resources? This is what Transfer Learning [16] was imagined for. From an initial domain and task, transfer learning uses knowledge acquired to learn a new task on a new domain, under the hypothesis that the two domain-task tandems are different but similar in some way, as knowledge earned on the first one should allow to solve the new one faster.

While various methods exist, this master’s thesis is based on parameter-based transfer learning. This method focuses on shared weights between the two models at hand. The idea is that this will allow the model to converge faster as initial weights should be close in value to the optimal weights, as otherwise weights are conventionally initialized at random. If the hypothesis is accurate, this method should indeed minimize calculation, as less epochs are needed to converge.

2.5 Grad-CAM

As the need for interpretability rises from the emergence of Artificial Intelligence (AI) application on critical domains, explainable AI (XAI) solutions are created to extract motive behind an AI output. For the peculiar case of CNN, Gradient-weighted Class Activation Mapping (Grad-CAM) [17] has been developed. From an image input and the name of the last convolution layer of a network, it is able to offer a visual representation via a 2D heatmap of the region(s) of the images

used to pick the label returned. As presented in the article [17], the heatmap is computed in three steps.

1. First, the feature maps of the convolutional layer are pooled with Global Average Pooling (GAP) and, for each class c , a class score y_c is extracted with a linear transformation. This function generates Class Activation Maps (CAM) and comes from another article [18].

$$y_c = \underbrace{\sum_{k \in K} w_k^c}_{Dense} \underbrace{\frac{1}{Z} \sum_{i=0}^n \sum_{j=0}^m A_{ij}^k}_{GAP}$$

where K is the set of feature maps, n the height of the image, m the width of the image, $Z = n * m$ the total number of pixels, A^k is k^{th} the feature map and w the weights of the dense layer.

2. Second, the gradient of y_c with respect to the feature map A is computed and another GAP is applied.

$$\alpha_k^c = \underbrace{\frac{1}{Z} \sum_{i=0}^n \sum_{j=0}^m}_{GAP} \underbrace{\frac{\delta y^c}{\delta A_{ij}^k}}_{Gradient}$$

this gives a new weight, α_k^c , representing the partial linearization of the network downstream from A .

3. The heatmap is finally generated by a weighted sum of the feature maps passed through a ReLU.

$$L_{Grad-CAM}^c = ReLU(\sum_k \alpha_k^c A^k)$$

All that is left to do before visualisation is normalisation between 0 and 1.

Chapter 3

State-of-the-art

With the rise of efficient computer vision tools, deep learning application in medical imaging have been a hot topic for researchers around the world. This section will explore the current advances in the field, starting by the status of application in radiology. The focus will then move on to existing tools for pneumonia classification using deep learning.

3.1 Deep Learning and Radiology

In the article *Deep Learning in Radiology*[19], authors offer an overview of the current status of Deep Learning (DL) tools in general and in healthcare. The tasks Deep Learning is able to undertake have to be specific. Deep Learning models trained on ImageNet[20] dataset exhibit particularly great results on chest images, which is surprising as ImageNet does not contain medical images. Machine Learning in general obtains convincing results in classification problems, and Deep Learning offers the additional advantage of discovering its own discrimination features in autonomy. In tasks for classification, chest imaging related task, CNNs have yielded to promising results. The main examples are: classification of lung nodules between malignant and benign categories, signaling whether endotracheal is present as well as diagnosis of tuberculosis and interstitial lung diseases. The article mentions a few current limitations, such as the fact that DL model sometimes makes mistakes that users may not expect. Undetectable to human eye alteration are able to render the tools inefficient. Obtaining sufficient and qualitative data as well as universally accepted annotations for training proved to be a difficult task. Legal and ethical questions have risen as to who should be held accountable for the interpretations of the models [19].

3.2 Pneumonia

Moving closer to the research question of this work, the following section will discuss what has been done in the domain of pneumonia classification. Two leading datasets are available in this field: NIH’s chestX-Ray14 dataset [21] and Chest X-Ray Images for Pneumonia Detection Dataset [22]. They will be discussed in detail in Chapter 4, at this point the main difference readers should be aware of is that the former focuses on adult X-Ray, while the later offers X-Ray of children between one and five years old.

3.2.1 NIH’s chestX-Ray8 dataset

NIH’s chestX-Ray8 dataset is the leading dataset for lung disease classification. In this first iteration, classification task for pneumonia achieves a 0.63 Area under the Receiver Operating Characteristic Curve (AUROC). The ROC is a graphical representation that is determined by the evolution of the True Positive Rate (TPR) with regards to the False Positive Rate (FPR) [23].

$$TPR = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$FPR = \frac{False\ Positive}{False\ Positive + True\ Negative}$$

AUC is bound between zero and one, and tends to one when the model is efficient. The article also attempted to extract heatmap highlighting the models feature of choice to offer human-level interpretability. It obtained a localisation accuracy of 75% [21].

CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning revisited the dataset using a DenseNet network and improved the AUROC to 0.7680. Interpretability feature was implemented using CAM, the predecessor of Grad-CAM. No evaluation the heatmap is offered [24].

3.2.2 Chest X-Ray Images for Pneumonia Detection Dataset

Chest X-Ray Images for Pneumonia Detection Dataset was published in an article offering identification of various diseases. Their initial model had a classification accuracy between pneumonia and non pneumonia patient of 92.8% [22]. The model does not offer any interpretability feature, rendering its usage on the field risky, as no human can ensure its efficiency.

In the article *Comparison and Validation of Deep Learning Models for the Diagnosis of Pneumonia*, authors attempted to compare various backbone, trained the model using transfer learning and concluded that the most efficient backbone was MobileNet [25].

The article *Visualization and Interpretation of Convolutional Neural Network Predictions in Detecting Pneumonia in Pediatric Chest Radiographs* [26] is the first one to use Grad-CAM on this dataset. The images feed to the CNN model are masked by segmenting lungs. No metric evaluating the resulting heatmap are available [26].

3.3 Objectives

Seeing the lack of evaluated interpretable CNN in the literature, this master's thesis sets off to implementing a CNN coupled with Grad-CAM heatmaps. Emphasis was put on the evaluation of the heatmaps in order to assess whether maps are efficient as well as keeping the tools used accessible for Open Source.

Chapter 4

Datasets

The first step in order to develop a Convolutional Neural Network (CNN) diagnosing pneumonia for children was to find a dataset of children X-Ray labelled for pneumonia. The interpretability could be done using image segmentation, but as no available annotated dataset were found at the time, the approach was changed to use the classifier itself to highlight regions of interest by using techniques from the field of explainable artificial intelligence (XAI). This might alleviate the constraints on the dataset, meaning it could only need to be labelled and have enough data for the model to be trained and tested on. That lead us to the Chest X-Ray Images for Pneumonia Detection Dataset [22] and its updated version [27], that is the dataset used to build the model. It is available under CC BY 4.0 [28]. Creative Common License may not be mentioned by the Open Source Initiative (OSI), but they are still compatible, as they simply were not analysed due to the fact that they are not designed for code, the field covered by OSI.

In order to test the possibility for knowledge transfer and, possibly, XAI, a secondary dataset was needed. This implied finding an annotated dataset with a licensing allowing the model publication through Open Source. The only suitable candidate was NIH's chestX-Ray14 dataset [21], available under CC0 license [29], which means the work is available in Public Domain. This dataset contains fifteen different categories (for fourteen diseases and patients where none of the 14 disease could be found). This approach had to be changed though, as will be discussed later.

4.1 Labels

The labels from the Chest X-Ray Images for Pneumonia Detection Dataset were used throughout all out experiments, including when using NIH's dataset for

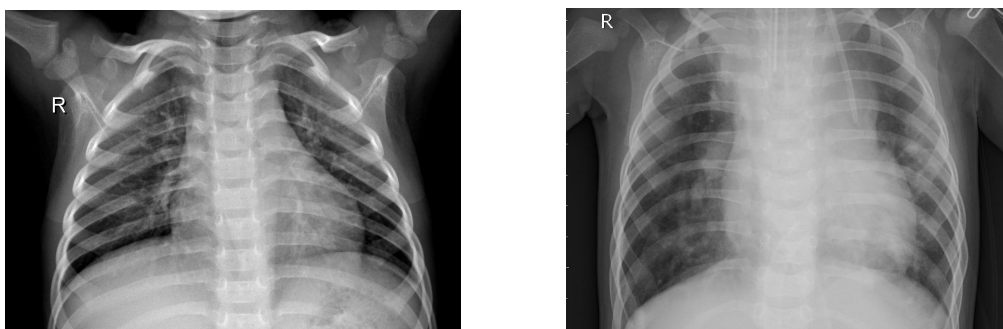
experiments. There are two categories:

- PNEUMONIA: positive images of patients diagnosed with pneumonia.
- NORMAL: negative images of patient diagnosed without pneumonia. However, as it will be discussed later, they may not be healthy either.

4.2 Data Exploration

4.2.1 Chest X-Ray Images for Pneumonia Detection Dataset

Chest X-Ray Images for Pneumonia Detection Dataset contains 5856 X-Ray images of pediatric patients from Guangzhou Women and Children’s Medical Center. The patients are between one to five years old. Not much information about the creation of the dataset could be found, and it was not annotated, which is why radiologists were asked to annotate and discuss it in Section 4.3.



(a) Sample X-ray labelled as NORMAL (b) Sample X-ray labelled as PNEUMONIA

Figure 4.1: Sample X-Rays from the Chest X-Ray Images for Pneumonia Detection Dataset. Patients here are children. The chosen pneumonia sample was validated as positive by the volunteer radiologist during the data collection described in Section 4.3.

Upon checking the initial sets made in the dataset, it was clear that the validation set was not sufficient (only 16 images from the total of 5856 images of the dataset). To prevent any further issues, a new validation set was derived from the initial training set. As multiple images of the same patient can be present, it was enforced that all images from a patient would have to be in the same set. Twenty percent of the patients from the training were extracted to create the new validation set. They were also removed from their original set to avoid redundancy and contamination

between sets. As the testing set already contained close to 10 percent of the set, it was kept as it was. Resulting set sizes are reported in Table 4.1.

	Training	Validation	Testing
Initial	5216	16	624
New version	4105	1111	624

Table 4.1: Set sizes before and after custom set separation for Chest X-Ray Images for Pneumonia Detection Dataset.

The dataset is further distinguished by two labels: **NORMAL** and **PNEUMONIA**, the former representing control images (patient not suffering from pneumonia) and the latter representing patient positive to pneumonia. In the initial set, both training and testing set present an imbalanced biased towards pneumonia cases while the validation set is balanced. After preprocessing, the training and validation set present a similar balance, while the testing set is around 10 % less imbalanced.

	Training	Validation	Testing
Initial	26%/74%	50%/50%	37%/63%
New version	26%/74%	24%/76%	37%/63%

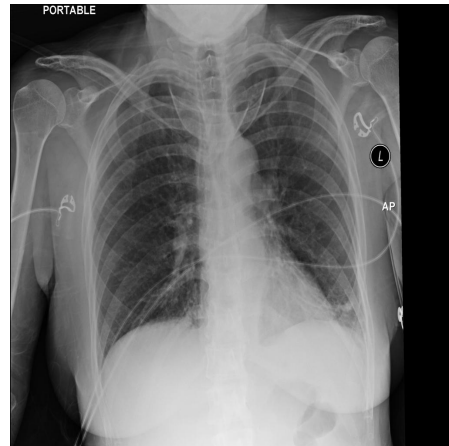
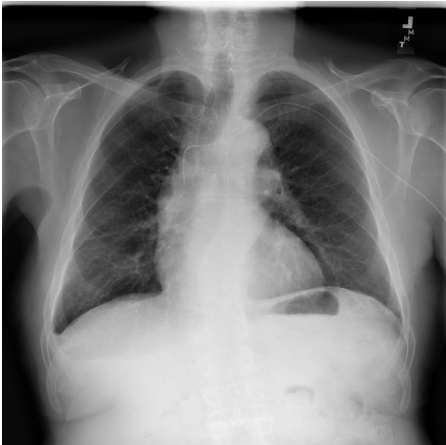
Table 4.2: Set label proportion (NORMAL/PNEUMONIA) before and after custom set separation for Chest X-Ray Images for Pneumonia Detection Dataset.

4.2.2 NIH’s chestX-Ray14 dataset

As knowledge transfer between datasets was tested in an experiment, a secondary dataset was required. Therefore, the NIH’s chestX-Ray14 dataset had to be analysed too. As the dataset is designed for 14 different lung diseases. This dataset, unlike the previous one, is not already separated in training, validation and testing. This meant the design of the sets was left to the user to create. Even though the present work is focused on pneumonia, it is interesting to acknowledge that the dataset offers 15 different labels for 14 diseases and healthy patients:

- Atelectasis;
- Cardiomegaly;
- Effusion;
- Infiltration;

- Mass;
- Nodule;
- Pneumonia;
- Pneumothorax;
- Consolidation;
- Edema;
- Emphysema;
- Fibrosis;
- Pleural_Thickening;
- Hernia
- No Findings.



(a) Sample X-ray labelled as NORMAL (b) Sample X-ray labelled as PNEUMONIA

Figure 4.2: Sample X-Rays from the NIH's chestX-Ray14 dataset. Patients here are adults. The samples were chosen from the second pool from the set building described here after.

The dataset comes with pdf files describing the dataset and its usage, as well as comma-separated values (csv) files describing each image individually. It is important to note that the tags were extracted from the report using Natural Language Processing (NLP). In other words, the radiology reports are analysed by a program designed to extract diagnosis from the text in the report. The specific tools, as found in the article publishing the dataset [21] are DNorm [30] and MetaMap [31]. In the description of the dataset, the accuracy is said to be estimated superior to 90 percent. Looking into the articles, precision and recall for each method can be found. Both metrics are computed using the confusion matrix as follows:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Seeing the results in Table 4.3, it appears that the annotation from this dataset cannot be fully trusted, while still probably mostly correct.

	Precision	Recall
DNorm	0.797	0.713
MetaMap	0.85	0.78

Table 4.3: Precision and Recall reported by their creators in their original articles for both NLP models used to annotate chestX-Ray14.

The training, validation and testing sets were created using data available in the comma-separated values files named `Data_Entry_2017.csv` and `BBox_List_2017.csv`. For each image of the dataset, the former has the following columns:

- **Finding Labels:** labels that were extracted by the NLP tools, separated by vertical bar symbol (`|`), as one image can present multiple findings;
- **Follow up #:** indicates the follow up it was taken at, with the first visit set to 0;
- **Patient ID:** unique ID of the patient;
- **Patient Age:** age of the patient at the time;
- **Patient Gender:** gender of the patient;
- **OriginalImage[Width:** width in pixels of the image;

- **Height**]: height in pixels of the image;
- **OriginalImagePixelSpacing**[**x**: scale to represent the real space between two pixels (in mm) over x-axis;
- **y**]: scale to represent the real space between two pixels (in mm) over y-axis.

Most of the columns can be linked to conventional field of the DICOM standard [32].

The latter file contains annotations for part of the available imaging. It offers the following columns:

- **Image Index**: name of the file containing the image, as to link the bounding box to its image;
- **Finding Label**: label of the annotation;
- **Bbox** [**x**: x coordinate (total number of pixels from the left side of the image) of the upper left corner;
- **y**: y coordinate (total number of pixels from the top side of the image) of the upper left corner;
- **w**: width in pixels of the rectangle starting from the point described by `bbox_x` and `bbox_y`;
- **h**]: height in pixels of the rectangle starting from the point described by `bbox_x` and `bbox_y`;

According to the authors' article [21], the bounding boxes are hand labelled, and can be used as ground truth. These images should thus be the ones used to test the Explainable AI part of this work. As a result, all of the testing set is taken from this set.

Two trios of ensemble (training, validation and testing) have been set up. In both of them, the set size was fixed by the number of pneumonia images available. As the other dataset has an imbalance towards pneumonia cases, while this one as whole offer much more images from other cases, it would not make sense to induce a strong opposite in balance when transferring knowledge. Any patient picked in one category was removed from the pool to avoid contamination. As the experiment required a smaller set of images the result from the previous set could transfer knowledge on, the choice focused on images presenting solely pneumonia,

to avoid a disease not appearing in the primary set poisoning the resulting deep learning network, as recent research have proven that inaccurate or deformed data could impact a Machine Learning performance [33]. The sets are designed to be balanced between labels.

The difference between the two pooling is that the first one filled the NORMAL labelled set with images reported as No Finding in the original data set. The second pool on the other hand, check the distribution between all the other disease and made a pooling of all of them according to this distribution. In the event of a label appearing too little for the ration to allow one image, one image from the No Finding group was removed to be replaced with one image of the rare label.

	Training	Validation	Testing
Pool 1	338	42	42
Pool 2	338	42	42

Table 4.4: Set sizes of set separation for NIH’s chestX-Ray14 dataset.

	Training	Validation	Testing
Pool 1	50%/50%	50%/50%	50%/50%
Pool 2	50%/50%	50%/50%	50%/50%

Table 4.5: Set label proportion (NORMAL/PNEUMONIA) for NIH’s chestX-Ray14 dataset.

4.3 Annotations

In order to analyse Grad-CAM performances, a ground truth was needed to compare results with. Many options were available such as finding a way to accurately evaluate on the annotated images of the NIH dataset (adult), using a tested detector to detect pathology signs or get viable annotation for the Chest X-Ray dataset. As described in Section 6, the latter was chosen by elimination. The radiology service of the Clinique Saint-Pierre Ottignies kindly accepted to annotate a few images and offered many insights and theories about the acquisition modalities and image quality.

4.3.1 Annotations Collection

The Chest X-Ray Images for Pneumonia Detection Dataset was annotated close to twice in total, once by Dr Arnould Marie-Cécile, radiologist, and a second partial

time by a tandem of assistant, Quentin Conroux and Amine Dinia. To avoid bias, the annotation from one participant were hidden to the others but one could access its annotation at any time. The assistant tandem was received one person at a time and each annotated roughly half of the dataset, without overlap.

The annotations were done using bounding boxes, which are rectangles with edges perpendicular to the image borders that contains the object of interest, here symptoms highlighting a pneumonia on X-Rays. These were recorded with makesense.ai [34], an Open Source labelling tool available under GPLv3 license [35]. This license is endorsed by the Open Source Initiative (OSI) and can be seen on their website [36]. This tool is free to use and does not store the data, two important point of this work as it aims to highlight Open Source and deals with sensitive data. Make Sense allow exportation of files containing the resulting data. One image can receive multiple boxes. The images loaded in the tool were extracted from the testing set to ensure realistic result by comparing results on images unseen in training.

Diagnostic screen could not be used due to technical difficulties and hospital cyber-security rules, so the images had to be displayed on a standard high-resolution screen. This means radiologist had a reduced resolution from their usual set up and they did not have access to any additional image transformation like flipping or inverting. A guide was sat next to them to supervise the process and answer any questions about the process or the objectives before, during and after the collection.

Participants were tasked with annotating one image per patient. The concept of bounding box was introduced to them, and they were asked to highlight symptoms or signs they would flag for a positive pneumonia diagnosis. It was also explained to them that they could add multiple boxes if the pneumonia was multi focal, one per focal point. As the time windows for annotation was short, one image was pre-selected for each patient, but the volunteers were encouraged to request to see additional pictures for patients associated with multiples X-Rays. In this case, they could pick any image in the selection to annotate on. They were told to ignore any patient they felt could not be diagnosed from the set of images available. Going back, erasing and modification were allowed, and no additional time constraint was added. They were free to stop at any time and the minimal number of patients to annotate was set as low as possible (twenty per person), as to give them plenty of time to analyse the data. The number twenty was chosen because, as we should have had four participants instead of three, it would have made a least 80 patients annotated, which was a sufficient number to be able to compute a representative metric of performance.

4.3.2 Discussion and insights given by healthcare professionals

The participants offered a lot of insights about the dataset while experimenting with it. One big point was the fact that the images were of poor quality due to visible circumstances. They described a lot of images as hard to read and potentially more difficult to accurately diagnose.

First off all, usually patients receiving chest x-rays are asked to take a deep breath and hold it during image acquisition to fully fill their lungs. This limits movement blur caused by breathing and makes the lungs and heart easier to observe because the ribs are open and lungs are expanded. In this case, many pictures are not taken with filled lung. Many factors could have caused this. Children between one and five may not be able to understand or follow instructions. This means the technician has to time their breathing to get the images, which only solves part of the problem (when it is managed) as this cannot prevent motion blur. Many of the children present acute symptoms of lung illness, which may cause laboured and irregular breathing pattern, making this task even more complicated.

A lot pictures are likely taken in intensive care, as many patients are intubated. In this case, the pictures are taken on patient lying down instead of standing due to their condition, in this case acute respiratory difficulties or failure due to pneumonia and other lung illnesses. This cause fluids to accumulate parallel to the image plane, in the back of the patient, instead of going down with regards to the lungs. This makes it harder to notice.

Some patients had to be skipped, and the participants wondered if the recorded diagnosis was done only with X-rays or the usual way, with medical imagery coupled with other clinical exams. Some patient were skipped by volunteers due to images quality or inability to find any sign of pneumonia on images. It was mentioned that sometimes clinical exam can diagnose pneumonia before it appears on X-rays. If the diagnosis was done using only clinical exams this may explain the difficulty encountered, but also may be a sign that these kinds of images should be removed from the set, given that presence of signs of pneumonia may be absent.

Explanation for the delay in apparition of visible symptoms on X-ray can be found by the developpement of the infection. Early signs included fever, general unease and minor tissue affection that may not appear on X-Ray while pulmonary affection may develop in two to three days.

Upon seeing the dataset offered the information of underlying cause of pneumonia was viral or bacterial, one volunteer added that while the illness can often be diagnosed thanks to medical imaging, this type of exam is not the leading factor for differentiation between the two.

4.3.3 Results

The results are stored in three comma-separated values (csv) files, one per volunteer. Every bounding box is represented by a row meaning multiple boxes appearing on the same image simply appear on multiple rows. The file has eight columns:

- **label_name**: label of the annotation, in this case it is always PNEUMONIA;
- **bbox_x**: x coordinate (total number of pixels from the left side of the image) of the upper left corner;
- **bbox_y**: y coordinate (total number of pixels from the top side of the image) of the upper left corner;
- **bbox_width**: width in pixels of the rectangle starting from the point described by **bbox_x** and **bbox_y**;
- **bbox_height**: height in pixels of the rectangle starting from the point described by **bbox_x** and **bbox_y**;
- **image_name**: name of the file containing the image, as to link the bounding box to its image;
- **image_width**: width in pixels of the image itself;
- **image_height**: height in pixels of the image itself.

We can easily compute all the corners of the rectangle by adding the width and height to the upper left corner coordinates, since the sides of the rectangle are known to be parallel to the image grid. In addition, the Python package *pandas* [37] [38] allows to easily load such files in a *pandas dataframe* to treat data easier. To highlight the possibility of creating such tool in Open Source, note that *pandas* package is published under The 3-Clause BSD License, which is endorsed by OSI [39].

In order to assess the consistency between volunteers, the annotated images of assistants are compared with the radiologist set.

	Radiologist	Assistant 1	Assistant 2
Number of images treated	174	40	56
Number of bounding boxes	198	64	71

Table 4.6: Number of images treated and boxes drawn per participant

	Radiologist & Assistant 1	Radiologist & Assistant 2
Mean	0.3001	0.1386
Maximum	0.5307	0.7526
Median	0.1018	0.2945

Table 4.7: Statistics about the IoU between Radiologist and Assistants over all common images.

The Intersection of Union (IoU) metrics will allow to see how much the annotations overlap between volunteers, which gives an idea about the consistency of reading X-ray images looking for pneumonia in the dataset. Its computation is detailed in Section 6.3.2. The results, available in Table 4.7, are fairly low. This could be explained by the quality of the X-Ray images, as explained by the volunteers, or human perception. On the other hand, it also means that the bounding boxes are often placed on the same areas of the pictures.

Chapter 5

Neural Network

This section will cover the building blocks of PneumonAI, an open source Convolutional Neural Network classifying X-Ray of children between one to five years old in two categories: pneumonia (positive for pneumonia) or normal (negative for pneumonia). It will discuss the implementation and results of the classifier, from preprocessing images of the dataset described in Chapter 4 to analysing their features to output a diagnosis. It will also test the possibility of transferring knowledge from one dataset to another by fine-tuning the network to a new dataset.

In an effort to prove the possibility of creating Open Source Deep Learning Tool, readers should be aware that the python packages used are Tensorflow [40] and Keras [15], both available under Apache License 2.0 [41]. This license is approved by the Open Source Initiative (OSI) and appears on their website [42].

5.1 Machine specification

All the metrics available in this section have been collected on the same machine with the following specification:

- Intel core i7 CPU, 20 cores, 5.6 GHz
- NVIDIA RTX 4070 Ti Super, 16GB of RAM
- 32 GB DDR5 RAM

The trainings were computed on the GPU thanks to a Windows Subsystem for Linux (WSL) and NVIDIA's Compute Unified Device Architecture (CUDA).

5.2 Preprocessing

To make computation execute in a reasonable amount of time, the pictures were resized to 512x512 pixels. This is done when loading the sets. To avoid distortion, the parameter `pad_to_aspect_ratio` from the Keras loader was set to `True`. It resizes the image to the desired image size (given with the argument `image_size`) while keeping its proportions, adding null pixels on the borders either at the sides or top and bottom to keep the initial ratio between height and width. This is necessary as proportion can be important in medical images.

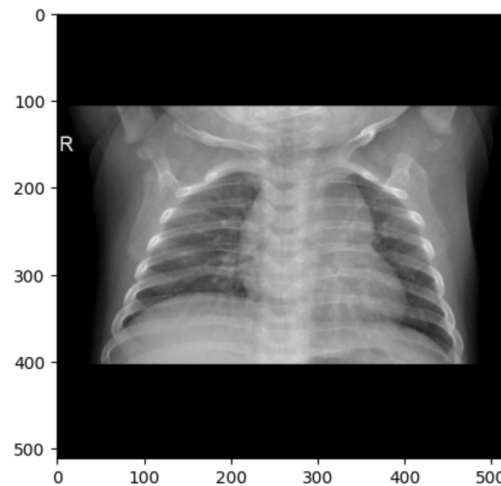


Figure 5.1: Sample preprocessed image.

As the final model used is MobileNetV2 [43], the applied preprocessing is chosen accordingly. In order to better understand the process, the source code was analysed. It turns out that it uses the preprocessing from the ImageNet[20] adaptation for keras in "tf mode", which is a of pixels scaling between -1 and 1.

5.3 Data Augmentation

The quantity of data available has to be sufficient to yield to sufficient data, peculiarly in healthcare [44]. One idea that arose was to create data from the available images. Data augmentation [45] is a method that applies this principle by applying various transformations at random on the original set to create more images. This method was used on both datasets in order to try to make the model more accurate in its prediction. When used properly, these techniques should allow to boost the quantity of data and minimize overfitting.

5.3.1 Image Random Transformation using `tf.image`

Tensorflow offers the `tf.image` module, that allows to apply such transformation on images without the need of an additional tool, since Tensorflow had already been elected for the model training by this point. The API offers 8 random stateless operations, which means that once a seed is fixed, the operation will always return the same result. This means any set creation is reproducible.

Data augmentation is applied after preprocessing and only to the training set. If it was applied to validation set, it could cause overfitting on feature caused by the technique and could go undetected in the testing set if it has been augmented too. It should not be used on the testing set in this case as we want to make sure it has a positive effect on the accuracy of real images.

In order to pick the methods to use, three of them were tested individually on a simplified model, then compared to a control training, using the plain dataset without any data augmentation. As the aim here was to improve the main model, this experiment was conducted only on the set of children's X-rays. The metric used for comparison is the training accuracy, as it is more sensitive to overfitting and will display lower results if the phenomenon appears.

The five untested functions are methods that did not make sense to apply in this case. The reasons are listed below:

- Vertical flip (up down flip): Applying a vertical flip would result in the lungs being upside down, which is to be avoided as it does not reflect an image that could be real [46];
- Hue: As the X-Rays are in gray scale, a hue change should not have an impact on the images;
- Saturation: As the X-Rays are in gray scale, saturation should always be zero impact on the images;
- JPEG quality: As the format of the initial images is not known, working on the JPEG quality encoding does not make sense;
- Crop: Randomly cropping the images to a fixed size might have resulted in images displaying more borders due to the resizing done in preprocessing than the X-Ray itself.

Which leaves the following methods :

- Brightness
- Horizontal flip (left right flip)
- Contrast

One might wonder why keep horizontal flips and not vertical flips, as the human body is not symmetrical, the most obvious argument being that the heart is on the left. There are known case of people living with their organs "flipped", and cases of dextrocardia [47], a congenital condition describing patient having their heart on the right side of their chest, were spotted in the testing set while annotating it with healthcare professionals.

Data Augmentation type	Training Accuracy
None (Control)	0.7965
Brightness	0.7901
Horizontal flip (left right flip)	0.8029
Contrast	0.8526
Contrast & Horizontal flip	0.8622

Table 5.1: Comparison of the training accuracy of the model depending on the data augmentation technique applied.

After consulting the results, available in Table 5.1, it was clear that contrast variations (an example can be seen on Figure 5.2) have a big impact on the training accuracy, while changes in brightness display results close to the control. Since horizontal flipping (an example can be seen on Figure 5.3) presented a slight improvement, an additional test was run combining it with a contrast related data augmentation.

The best combination appears to be the combination of the two best individual techniques. These are the operations that were applied on the final version of the model.

5.3.2 Noise

Another common data augmentation technique is to add noise. This was tested too using `tf.random` module. Various random noises following a distribution (for example, Gaussian and Uniform) were generated and summed to the images,

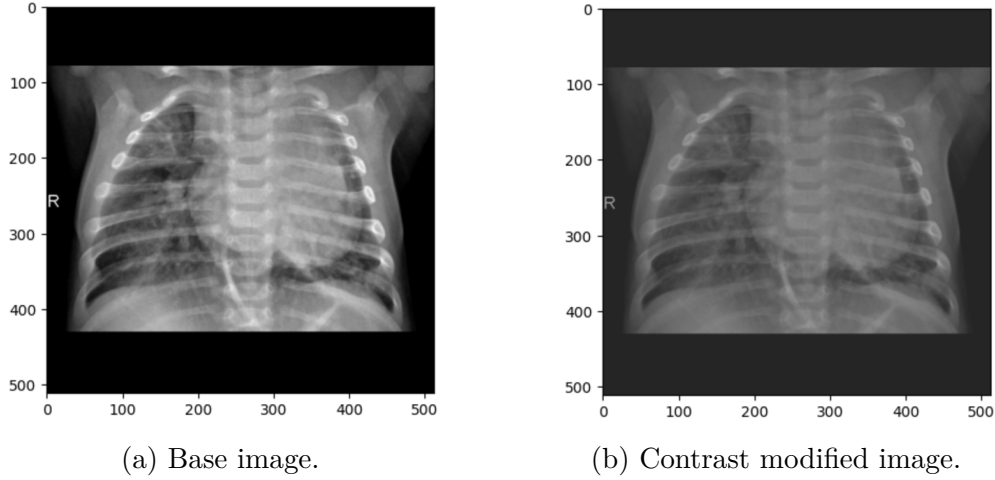


Figure 5.2: Example of contrast variation.

then clipped to stay within the bounds induced by the scaling applied during preprocessing.

Following the same *modus operandi*, the various noises were added to the already augmented training set and, after fitting the model, the training accuracy was checked. The accuracy was either unchanged or lower. Upon looking further, the resulting model exhibited overfitting in all of the test runs. Therefore, this method was discarded from the project.

5.4 Architecture

The model was built using parameter-based transfer learning. More precisely, the first layers are taken from an existing model (the backbone) and additional layers are added. Thanks to Tensorflow [40], loading a backbone and its weights is fairly easy. Indeed, `keras.application`, a Keras [15] module offers a generous range of pre-trained models easily loadable from python code, that can be injected in models fairly easily.

Backbones are loaded without their output layer, as it is related to the original task and thus most likely not adapted to the task the knowledge is transferred to. For this reason, additional layers extracting prediction from the feature extracted by the backbone, and an output layer allowing to output prediction from the classifier. The architecture is schematised on Figure 5.4, and features three additional layers: a 2D GAP, a dropout layer and finally, a dense layer. The last layer of PneumonAI

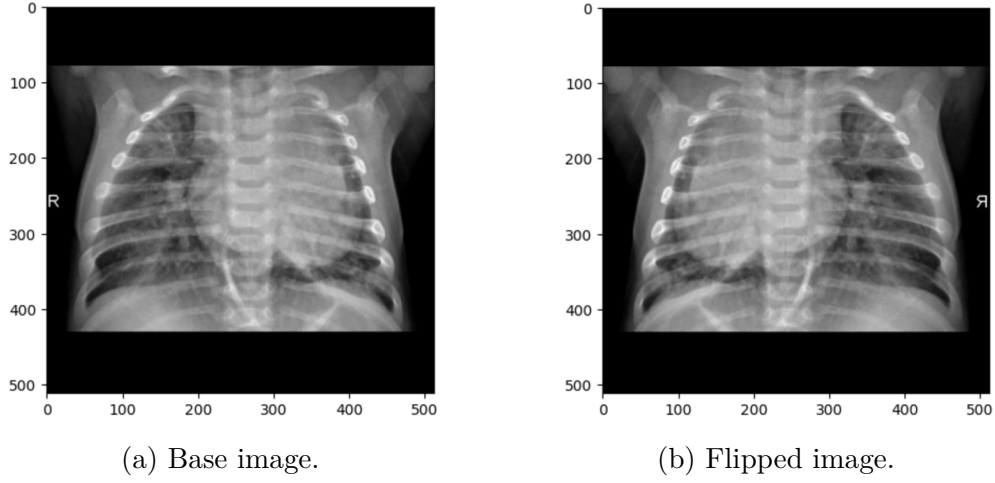


Figure 5.3: Example of horizontal flipping.

has an output of size one, and summarises all the features from the previous layer in one prediction. The output size is directly linked to the number of labels. Since the task at hand is a binary classification, only one output is needed.

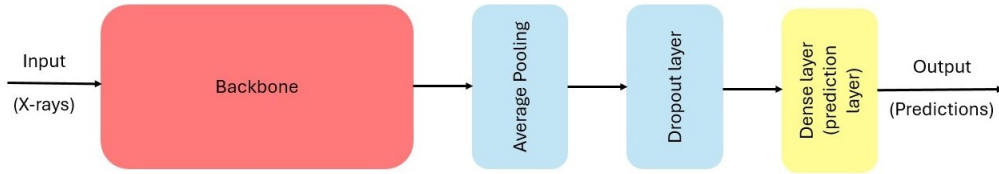


Figure 5.4: Schema of the architecture of the network. Note that the inputted X-rays are preprocessed according to the input expected by the backbone, to ensure compatibility.

5.4.1 Backbone

According to the literature [25], by comparing 4 different backbone (MobileNet, ResNet-18, ResNet-50 and VGG19), MobileNet exhibits the best result in accurately classifying the dataset at hand. Looking further, Xception was tested too, but gave slightly inferior results to MobileNet.

The article does not specify the version of MobileNet used. From the publication date, MobileNetV2 would be the most likely answer. By testing on the control machine, MobileNetV3, Small or Large, obtains results much poorer than its

predecessor, MobileNetV2. The testing accuracy ends up being 62.5% for both third generation pre-trained models against 86.2% for their ancestor. The newer generation display higher validation (around 75%) and training accuracy above 90% despite their poor performance on testing set, which may be a sign of overfitting. With that in mind, the backbone chosen was MobileNetV2 [43].

The architecture of MobileNetV2 [43] starts with a fully convolutional layer chain utilising 32 filters, followed by 19 building similar blocks. Convolution operations are done using 3x3 kernels. A batch normalization layer is included between each block, either as last layer or before the add layer. There are two main types of blocks, differentiated by a stride number of one or two in the original article. The authors introduced a bottleneck depth-separable convolution with residuals, that compose 16 out of the 19 blocks of the chain. The schema the authors made to present these blocks is reported by Figure 5.5.

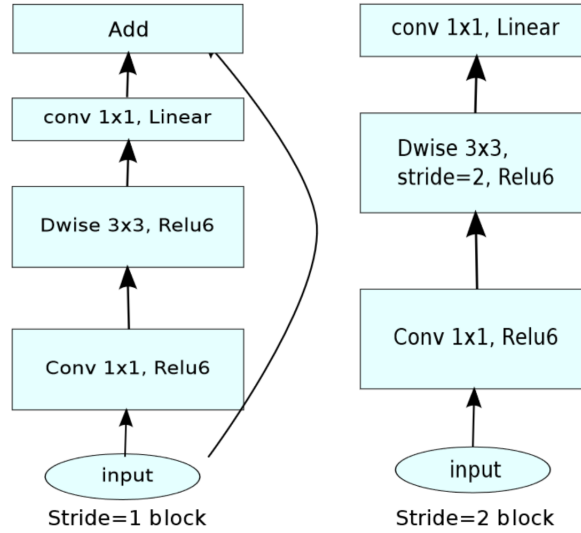


Figure 5.5: Schema describing the convolutional blocks in MobileNetV2, depending on the stride, directly taken from the authors' article [43].

The chain starts with a 2D convolution block, followed by 16 custom bottleneck blocks. Another 2D convolution block is then applied.

By this point, only 18 out of the 19 blocks were covered. The last block is used to output the result of the classification and is thus not loaded in this implementation.

Block number(s)	Stride
1	1
2 to 3	2
4 to 6	2
7 to 10	2
11 to 13	1
14 to 16	2
16	1

Table 5.2: Stride of the 16 bottleneck blocks.

5.4.2 Complete Network

Associating the backbone to the additional layers gives us the complete network containing 2,261,829 parameters, making up for 8.63 MB of memory usage. The network itself makes up for 2,259,265 parameters, including parameters being added by Adam Optimizer, that defines the gradient descent used to compute weights following Adam algorithm [48].

5.5 Training

The training is divided into two steps: the training of the new outer layers, and the fine-tuning of MobileNetV2 to the target task. Both phases use Adam Optimizer and BCE loss described in Chapter 2.

5.5.1 Outer layer training

When the model is initially built, the shared weights brought by parameter-based transfer learning should be relatively close to the target of the gradient descent on the loss. On the other hand, the additional layer created to transfer to the target domain and task have been initialised at random, the probability of these being far from their optimal value is high. In order to save computing power, we freeze the weights of the backbone. This means that their values are fixed and not considered during the gradient descent. This allows to reduce the number of parameters to train from more than two million to 1,281.

The network is then trained until a sign of convergence is detected. This is done using the callback module from keras: it allows to induce a condition based on chosen metric(s) that triggers the end of the fitting before the initial maximal number of epochs is reached. Thanks to the Early Stopping, the network training

is set to stop if validation loss stops decreasing for three epochs. This peculiar metric is chosen because the loss is the objective function of the problem, it is the convergence that should be monitored. Focusing on validation loss instead of training loss reduces risk of overfitting as the set used for its computation is not the training set.

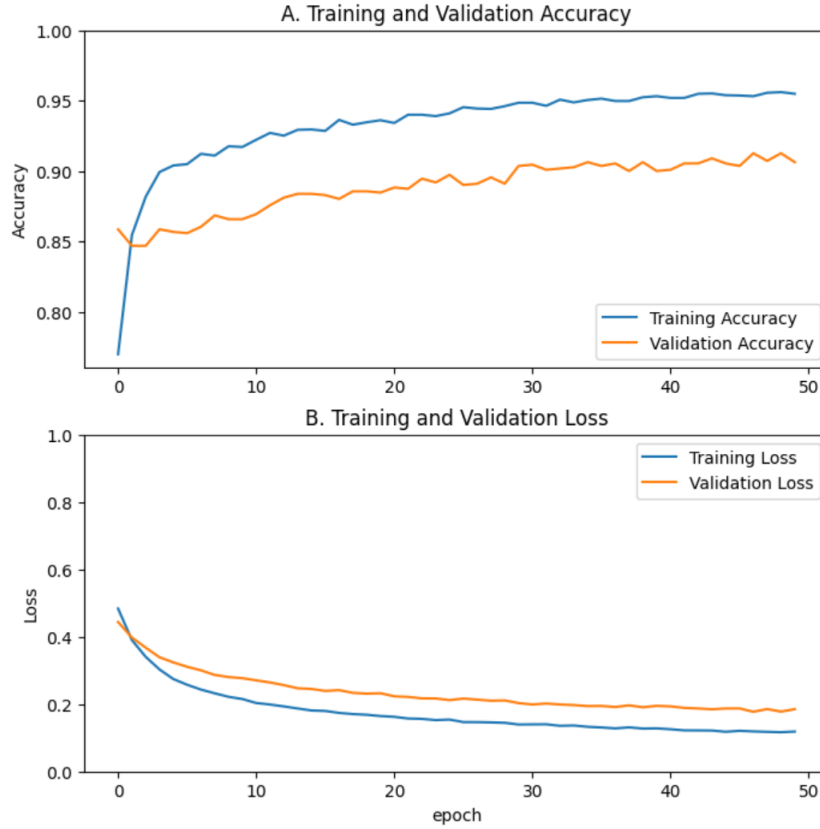


Figure 5.6: **A.**: Evolution of the training and validation accuracy per epoch during the training process. **B.**: Evolution of the training and validation loss per epoch during the training process.

The validation loss (see Figure 5.6) converges after roughly 50 epochs (usually between 45 and 55 epochs). The initial accuracy of each set (reported in Table 5.3) is fairly close to its set balance. This indicates poor results as a model reporting all cases as positive would have similar results. The trained network exhibits much better result, with an 86% training accuracy

	Training	Validation	Test
Initial accuracy	73.20%	75.25%	62.02%
Trained accuracy	95.49%	90.64%	86.86%
Initial loss	0.5710	0.5384	0.6993
Trained loss	0.1185	0.1850	0.3503

Table 5.3: Initial and trained network the accuracy of each set.

5.5.2 Unfreezing the backbone and fine-tuning

In order to refine the model, the backbone was then unfrozen. At this point, the weights should be fairly correct and thus little modification is needed. This allows to compute on the millions of parameters at the same time. An additional call back is added: ReduceLROnPlateau. It was set up to reduce the learning rate by dividing it by 10 whenever the network would converge, until the learning rate reached 10^{-6} . The initial learning rate model creation was of 10^{-4} . The patience of the early stopping was raised to 10 epochs in order to allow the new callback to trigger.

Looking at the results in Table 5.4, the model does not get any better. It actually starts to overfit over both training and validation set while performance of the testing set deteriorates. The loss oscillates around the equilibrium already found in the first phase of training (see Figure 5.7) for 67 epochs with a slight improvement on data known to the model. This could be a sign that the assumption on the shared weights with MobileNetV2 was accurate.

	Training	Validation	Test
Trained accuracy	95.49%	90.64%	86.86%
Fine-tuned accuracy	96.39%	91.63%	86.38%
Trained loss	0.1185	0.1850	0.3503
Fine-tuned loss	0.0929	0.1582	0.3511

Table 5.4: Trained and fine-tuned accuracy for each set.

5.6 Knowledge transfer between datasets

In order to evaluate the possibility of a knowledge transfer for one dataset to the other, an additional experiment was done using NIH’s ChestX-Ray14 dataset (the target dataset). The idea was to compare two experimental models (one trained

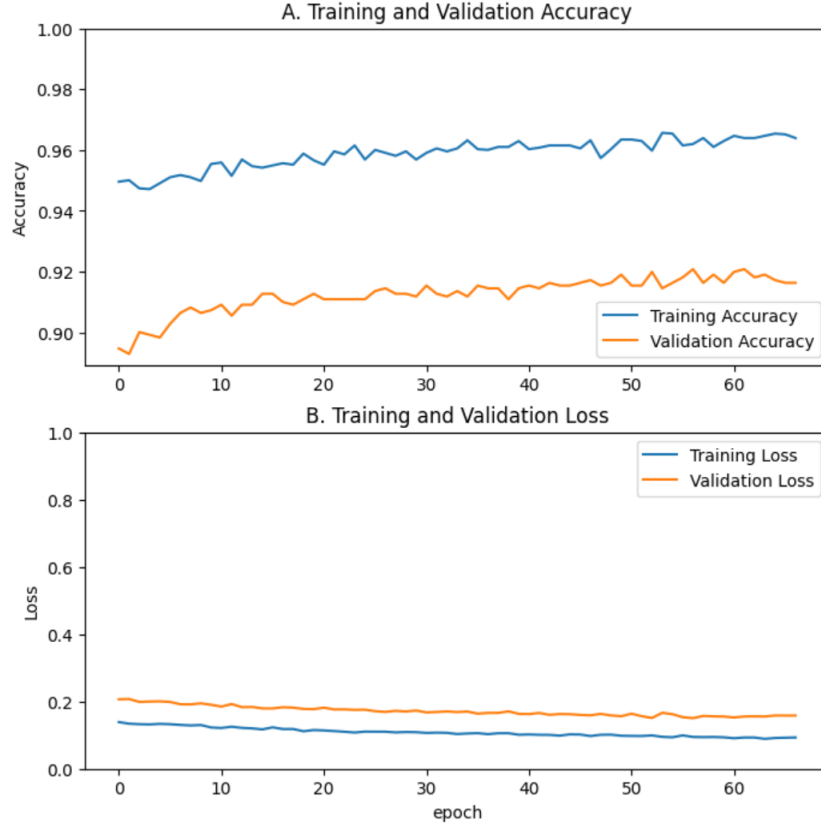


Figure 5.7: **A.**: Evolution of the training and validation accuracy per epoch during the fine-tuning process. **B.**: Evolution of the training and validation loss per epoch during the fine-tuning process.

with the initial dataset and fine-tuned on the target dataset, one the other way around) with a model fully trained on the target dataset. The experiment was repeated on the two trios of ensembles described in Section 4.2.2. Two different ensembles allow to adapt the target set to the source set, as the content of the NORMAL set of the Chest X-Ray Images for Pneumonia Detection Dataset was not described.

The results are reported in Table 5.5 and 5.6. In both cases, the model built without Chest X-Ray Images for Pneumonia Detection Dataset achieves the best results.

Trained on	Fine-tuned on	Training acc.	Validation acc.	Testing acc.
Target	Initial	52.51%	59.52%	51.19%
Initial	Target	68.34%	53.57%	57.14%
Target	Target	70.41%	52.38%	71.42%

Table 5.5: Accuracy on the target set of the model depending on the sets used for training and fine-tuning. The initial set is the Chest X-Ray Images for Pneumonia Detection Dataset and the target set is the trios of ensemble described in Section 4.2.2 containing only patients under No Findings label in the NORMAL set.

Trained on	Fine-tuned on	Training acc.	Validation acc.	Testing acc.
Target	Initial	51.77%	60.71%	51.19%
Initial	Target	65.89%	55.95%	54.76%
Target	Target	66.12%	52.38%	64.28%

Table 5.6: Accuracy on the target set of the model depending on the sets used for training and fine-tuning. The initial set is the Chest X-Ray Images for Pneumonia Detection Dataset and the target set is the trios of ensemble described in Section 4.2.2 containing all types of non-pneumonia patients in the NORMAL set.

5.6.1 Discussion and future work

As the results of the knowledge transfer are not convincing, the question of the possibility of a one-fits-all model for pneumonia detection is raised. Many plausible causes for these results can be found: the age of the patient, the image quality, modalities of image acquisition, X-Ray machine used, ... All of these parameters can vary in hospitals, which leads to the hypothesis that models could be more efficient if trained on images from its target hospital or even its target service. As Dr. Arnould mentioned in the discussion reported in Section 4.3.2 that modalities may vary between normal consultation and intensive care. Possibilities of using transfer learning to accelerate training would then become a turning point for future research, as each hospital could require a new training without much computing power at hand.

Chapter 6

Explainable AI (XAI)

As healthcare has many safety measures to ensure patients secure diagnosis, a model should be interpretable in order to be used on the field. Due to the impossibility to find a sufficiently annotated dataset to implement a CNN used for segmentation, the possibility of using Explainable AI (XAI) to highlight Region Of Interest (ROIs) could be the solution to allow development of such tools without needing fully annotated datasets. This section will cover the implementation and testing of an Explainable AI on the network trained in Chapter 5. Its evaluation will be done by comparing its result with annotations on the dataset done by the volunteers as described in Section 4.3.

6.1 Technology Choice

Articles on the secondary dataset have successfully implementing predecessors of Grad-CAM to help their model to be interpretable by humans [21]. With that in mind, Grad-CAM seemed like the best option in the domain of XAI.

6.2 Methodology

In order to prove efficiency of Grad-CAM, a ground truth was needed to compare results. Within the literature, the only annotated dataset about pneumonia is NIH's chest X-Ray14. From the experiment in Section 5.6, it is known that PneumonAI does not generalise to this dataset. Another lead could have been to look for an existing segmentation model. By logic, the evaluation of such a tool would have needed an annotated dataset. Once again, from trial and errors, this work proved that generalisation between dataset was not guaranteed, and knowing that the dataset has no annotation, such a tool would have had to be trained on another dataset. This left no other option than collecting annotations. The annotations

described in Section 4.3 have thus been used as ground truth to evaluate the heatmaps returned by Grad-CAM.

6.3 Metric

The metric used is the Intersection over Union (IoU). It has been a standard in the evaluation of bounding boxes, including in the domain of pneumonia detection as it has been used on the secondary dataset to evaluate Grad-CAM performances in its publication article [21]. Therefore, it makes sense to use it for the primary dataset as well, in order to have comparable results.

6.3.1 Extraction of the Bounding Boxes

As explained in Section 2.5, Grad-CAM returns a heatmap. In order to compare it with bounding boxes, a common format must be extracted from one way or the other. This issue was solved with a four steps process.

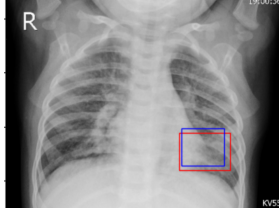
1. First, the heatmap is binarised using a threshold. Note that the heatmap is normalised between 0 and 1, so any threshold not contained in this range would result in a single category heatmap. Any pixel with a value above the threshold should be present in the final bounding boxes.
2. The positive pixels are clustered together with an algorithm inspired by region growing algorithm. The program will draw an unclustered pixel (the seed) and check its neighbor pixels. If any are positive, they will be clustered with the initial pixel and their neighbor will be tested recursively. Once no positive neighbor can be found anymore, another unclustered pixel is drawn to create a new cluster, until all positive samples belong to a cluster.
3. Then, each cluster is assigned a bounding box based on the coordinates of their up-most, bottom-most, left-most and right-most pixel.
4. Grad-CAM delivers a smaller matrix than the actual image, of 16x16 pixels, in order to move the coordinates back to 512x512, the following operation is applied:

$$\begin{aligned}
 x &= x^{Grad-CAM} * 32 \\
 y &= y^{Grad-CAM} * 32 \\
 x + w &= (x^{Grad-CAM} + w^{Grad-CAM}) * 32 + 32
 \end{aligned}$$

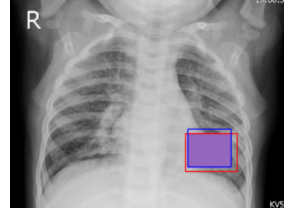
$$y + h = (y^{Grad-CAM} + h^{Grad-CAM}) * 32 + 32$$

where x and y are the coordinates of the top left corner, w is the width, h is the height and $i^{Grad-CAM}$ is the value of variable i returned by the extraction from the Grad-CAM heatmap.

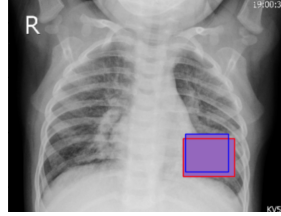
6.3.2 Computation of the IoU



(a) Sample boxes, sampled from the annotations collected.



(b) Intersection between boxes is highlighted in purple.



(c) Union between boxes is highlighted in purple.

Figure 6.1: Visual representation of the areas used to compute IoU. The red and blue box are the two boxes the IoU is computed on.

The Intersection of Union (IoU) is computed as such:

$$IoU = \frac{Area\ of\ Intersection}{Area\ of\ Union}$$

But this is done on two boxes only. In this example the number of boxes for one image may be different between the two annotations compared, and it is tedious to link boxes from both set in a one-to-one mapping. To solve this issue, the computations are done on all boxes from the image: the intersection is computed between each pair, and the area is the sum of all boxes area. It will also give a higher score if both parties notice multiple focal points. The function was implemented

by computing the sum of the intersection on each possible pair:

$$I = \sum_{i \in \text{Radiologist set}, j \in \text{Assistant set}} \text{Intersection between } i \text{ and } j$$

Which is then divided by the sum of the area of all the boxes minus the previously computed sum of intersections.

$$\frac{I}{(\sum_{i \in \text{Radiologist set}} \text{Area of } i) + (\sum_{j \in \text{Assistant set}} \text{Area of } j) - I}$$

6.4 Treatment of the annotations

Any comparison should be done based on common grounds. Grad-CAM will return a heatmap on the images feed to the CNN, meaning that the images would have undergone preprocessing already. On the other hand, the annotations were done on the raw images. In order to have compatible coordinates, one or the other must be resized. Applying scaling is fairly easy, as it does not affect bounding boxes, but resizing operation will affect the coordinates and dimensions of the bounding boxes. Thanks to `tf-image`, a package inspired by `tf.image` module, the resizing applied to images by TensorFlow can be reproduced on bounding boxes. The package is published under MIT license, which is endorsed by OSI[49] and does fit the Open Source criterion imposed on this work.

6.5 Results

The experiment was run on each annotation set. The threshold value was set to 0.5, 0.25 and 0.1. All the results are reported in Table 6.1. Compared to the IoU between radiologist computed in Section 4.3, it is low. Two explanations could be theorised: it could mean that the features are not meaningful for humans, and therefore not interpretable, or that the model did not study the same features as radiologists. In order to assess between these two possibilities, the heatmaps were analysed in Section 6.6.

6.6 Heatmap Analysis

Seeing Table 6.1 and with the insights received in Section 4.3.2, the heatmaps for positive examples were analysed by hand in order to understand what the model was looking for in images. The objective here was to understand why the resulting bounding boxes would barely overlap with the ones created by healthcare

	Radiologist	Assistant 1	Assistant 2
Threshold = 0.5			
Mean	0.0165	0.0329	0.0411
Median	0.0083	0.0215	0.0259
Max	0.3119	0.1606	0.2063
Threshold = 0.25			
Mean	0.0217	0.0362	0.0531
Median	0.0164	0.0280	0.0427
Max	0.1248	0.1126	0.1922
Threshold = 0.1			
Mean	0.0222	0.0383	0.0559
Median	0.0184	0.0301	0.0428
Max	0.0929	0.0902	0.1730

Table 6.1: Statistics about the IoU between volunteers and Grad-CAM over all common images depending on threshold.

professionals. It could be that the model was not interpretable, or simply that other significant elements allowed the model to efficiently discriminate between the two datasets categories.

On many images, hot spots can be noticed on the sides of the chest, highlighting the background between the arm and the chest, sometimes accompanied by a hot spot above the shoulder (see examples on Figure 6.3). At first glance, it may look like the result does not make sense, but given the insights from Section 4.3.2 and comparing the image with ones from the training set, one theory arises. In images labelled NORMAL, most children have their arms raised above their head (see Figure 6.2). It could be that the model uses arm angle to figure modalities of the images, which, in this dataset, is correlated with the label of the image.

Medical equipment such as electrodes (see Figure 6.4) and catheters (see Figure 6.5) attract hot spots as well. Following the same logic, the normal set was browsed and no sign of catheters or electrodes were found, meaning it could be an efficient way to discriminate in this particular dataset as well.

6.7 Discussion and Future Work

From the conclusion of the previous section, one could argue that Grad-CAM heatmaps do accurately explain the model choices, and that its reasoning is

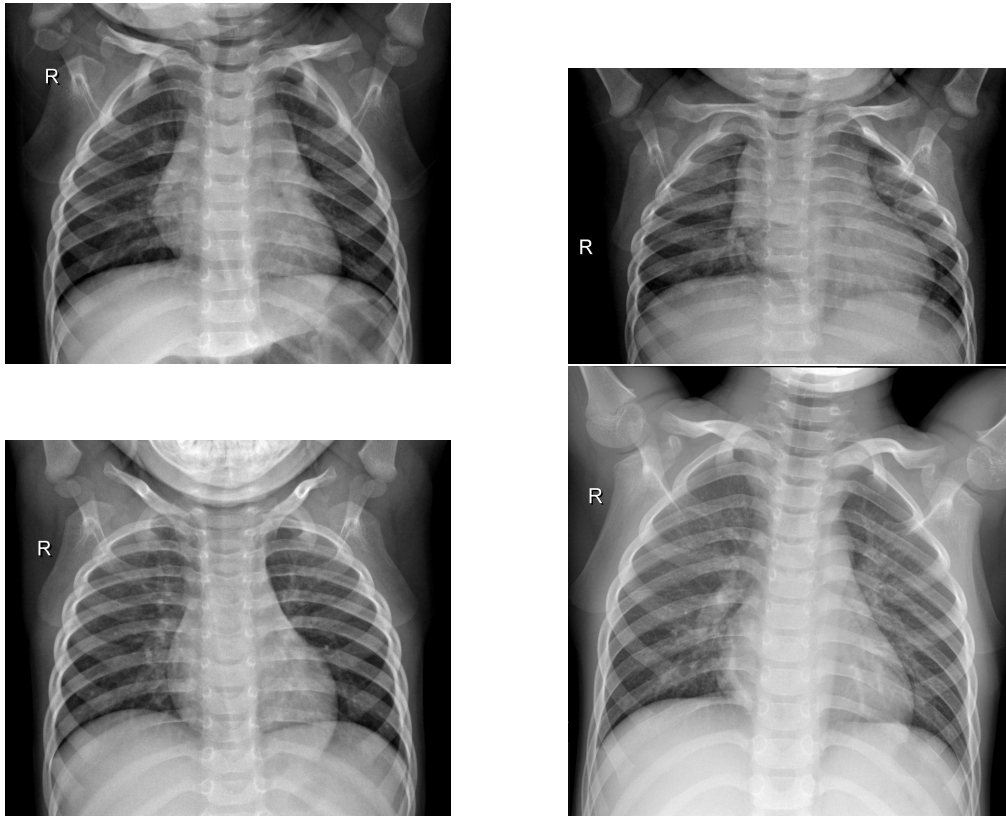


Figure 6.2: Sample images labelled as NORMAL

interpretable. In a way, it uses clinical signs of the disease to diagnose instead of infectious outbreaks. As it works fairly well on this dataset, the model can rely on these features in these conditions but this would not necessarily work on the field. To investigate this further and possibly put forward a better version of the model, a new dataset would be needed. This new version should be carefully selected so that visible signs translating modalities of the image acquisition do not allow easier classification than infectious outbreak.

In this case, even if its results are not groundbreaking, Grad-CAM did a good job at highlighting the AI discriminant features in a way that is easy grasp for humans.

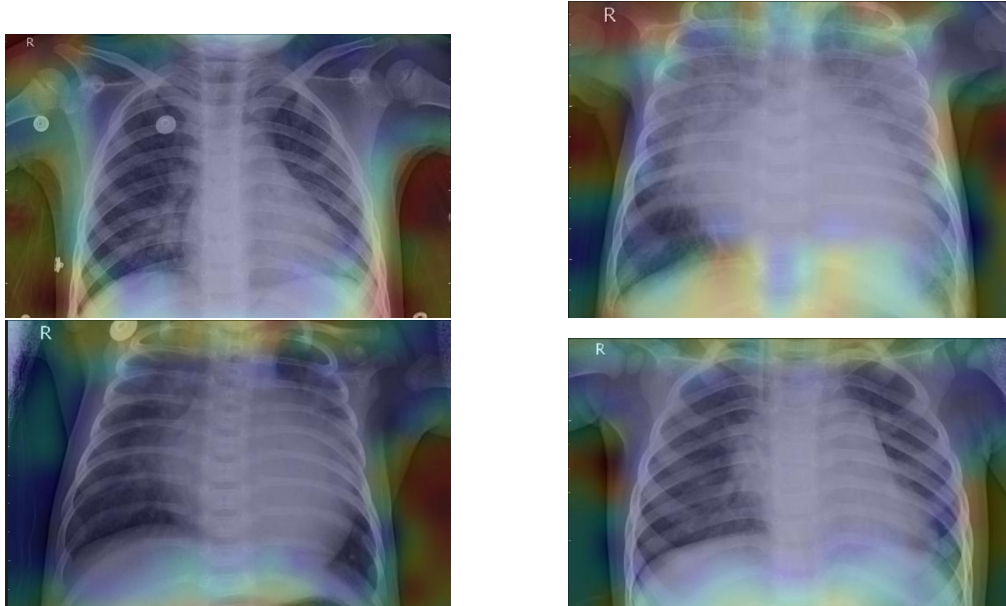


Figure 6.3: Images visualisation with the heatmap surimposed. The selected images exhibit hot spots between the arm and chest of the patient.

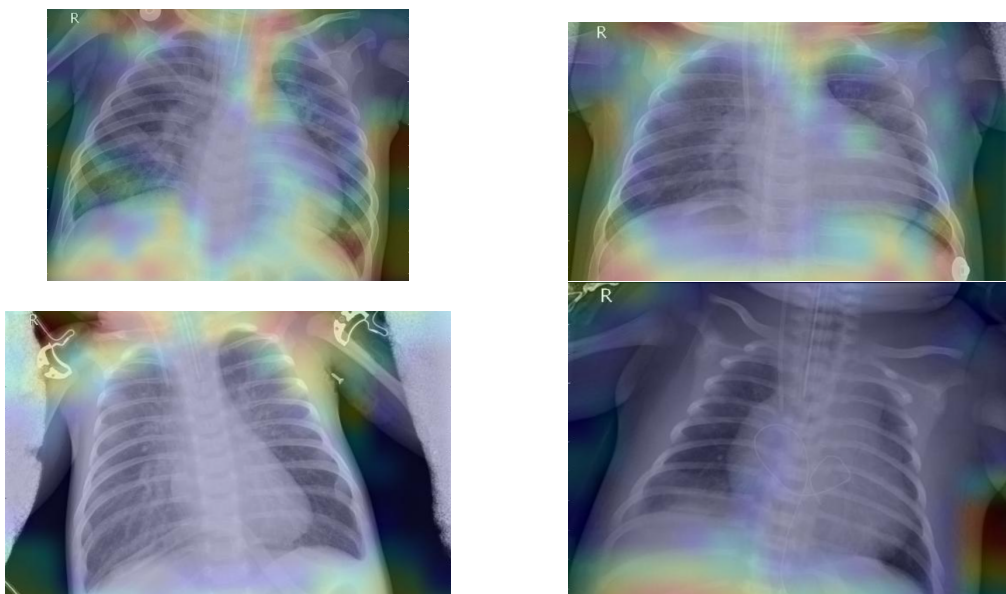


Figure 6.4: Images visualisation with the heatmap surimposed. The selected images exhibit hot spots on electrodes.

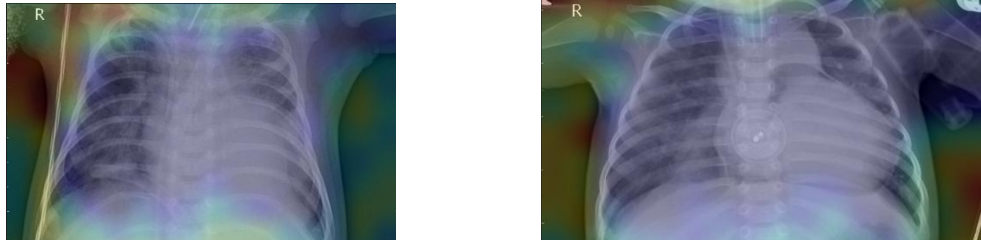


Figure 6.5: Images visualisation with the heatmap surimposed. The selected images exhibit hot spots on a catheter and a tube.

Chapter 7

Conclusion

Through this manuscript, an interpretable Open Source CNN diagnosing pneumonia on children between one to five years old using chest X-ray images was developed: PneumonAI. The whole process, from identifying the research question to building and evaluating the tool has been covered.

By exploring the dataset and discussing with healthcare professionals, hypothesis about images acquisition could be established and annotations of ground truth were collected. Training of a CNN was accelerated thanks to transfer learning, and efficient generalisation of the MobileNet to the task could be assumed seeing that unfreezing the backbone does not yield to an increase in performances. Knowledge transfer from a dataset of pediatric images to adult X-Rays proved to be inefficient. Finally, thanks to the input of healthcare professionals, an analysis of Grad-CAM heatmaps result enable to highlight the importance of dataset design for the creation of efficient DL models.

The process has only used Open Source friendly datasets, framework and libraries. The functions implemented during development have been made publicly available on the author's GitHub[50] under MIT license[49]. The accuracy of the model rose to 86.86%, which is only 5.98% away from the state-of-the-art on the used dataset. The work offered the first evaluated interpretability module on this dataset.

The result of Grad-CAM are not necessarily incorrect, as discussed in the previous chapter. The most common features highlighted appear to make sense in the dataset at hand, which means that creating a dataset that does not allow a model to accurately classify with clinical features. Instead, it could only display X-Rays taken with the same modalities could allows to improve its performances.

As seen in the article *Visualization and Interpretation of Convolutional Neural Network Predictions in Detecting Pneumonia in Pediatric Chest Radiographs* [51], a lead to improve Grad-CAM results could be to mask the image to only display lungs of patients to the CNN. This method would mean an additional model would be needed to segment the lungs. The existing model used for lung segmentation [52] in this particular article has been trained on another dataset. Seeing the resulting mask, cardiac pouch is masked on the X-Rays while it was often used by volunteer healthcare professionals during annotation collection described in Section 4.3. From new annotation, a segmentation of lungs including the heart could allow to redirect models feature choice to ones actually related to X-Ray reading.

Bibliography

- [1] U. A. Hôpitaux, “A20-01 iml-zmb - rapport audit imagerie médicale lourde,” tech. rep., INAMI, 2020.
- [2] P. M. Cheng, E. Montagnon, R. Yamashita, I. Pan, A. Cadrin-Chênevert, F. Perdigón Romero, G. Chartrand, S. Kadoury, and A. Tang, “Deep learning: an update for radiologists,” *Radiographics*, vol. 41, no. 5, pp. 1427–1445, 2021.
- [3] X. Liu, L. Faes, A. U. Kale, S. K. Wagner, D. J. Fu, A. Bruynseels, T. Mahendiran, G. Moraes, M. Shamdas, C. Kern, *et al.*, “A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: a systematic review and meta-analysis,” *The lancet digital health*, vol. 1, no. 6, pp. e271–e297, 2019.
- [4] WHO, “Pneumonia in children.” <https://www.who.int/news-room/fact-sheets/detail/pneumonia> [Accessed: 2024/05/30].
- [5] Y. Xu and R. Goodacre, “On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning,” *Journal of analysis and testing*, vol. 2, no. 3, pp. 249–262, 2018.
- [6] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [7] T. Yu and H. Zhu, “Hyper-parameter optimization: A review of algorithms and applications,” *arXiv preprint arXiv:2003.05689*, 2020.
- [8] S. Ruder, “An overview of gradient descent optimization algorithms,” *arXiv preprint arXiv:1609.04747*, 2016.
- [9] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

- [10] U. Ruby and V. Yendapalli, "Binary cross entropy with deep learning technique for image classification," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 10, 2020.
- [11] D. Bhatt, C. Patel, H. Talsania, J. Patel, R. Vaghela, S. Pandya, K. Modi, and H. Ghayvat, "Cnn variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, 2021.
- [12] B. Jähne, *Digital image processing*. Springer Science & Business Media, 2005.
- [13] T.-Y. Hsiao, Y.-C. Chang, H.-H. Chou, and C.-T. Chiu, "Filter-based deep-compression with global average pooling for convolutional networks," *Journal of Systems Architecture*, vol. 95, pp. 9–18, 2019.
- [14] P. Baldi and P. J. Sadowski, "Understanding dropout," *Advances in neural information processing systems*, vol. 26, 2013.
- [15] F. Chollet *et al.*, "Keras." <https://keras.io>, 2015.
- [16] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, pp. 1–40, 2016.
- [17] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that?," *arXiv preprint arXiv:1611.07450*, 2016.
- [18] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.
- [19] M. P. McBee, O. A. Awan, A. T. Colucci, C. W. Ghobadi, N. Kadom, A. P. Kansagra, S. Tridandapani, and W. F. Auffermann, "Deep learning in radiology," *Academic radiology*, vol. 25, no. 11, pp. 1472–1480, 2018.
- [20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.
- [21] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. Summers, "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3462–3471, 2017.

- [22] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, A. McKeown, G. Yang, X. Wu, F. Yan, *et al.*, “Identifying medical diagnoses and treatable diseases by image-based deep learning,” *cell*, vol. 172, no. 5, pp. 1122–1131, 2018.
- [23] K. H. Zou, A. J. O’Malley, and L. Mauri, “Receiver-operating characteristic analysis for evaluating diagnostic tests and predictive models,” *Circulation*, vol. 115, no. 5, pp. 654–657, 2007.
- [24] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, *et al.*, “Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning,” *arXiv preprint arXiv:1711.05225*, 2017.
- [25] Z. Yue, L. Ma, and R. Zhang, “Comparison and validation of deep learning models for the diagnosis of pneumonia,” *Computational intelligence and neuroscience*, vol. 2020, 2020.
- [26] S. Rajaraman, S. Candemir, I. Kim, G. Thoma, and S. Antani, “Visualization and interpretation of convolutional neural network predictions in detecting pneumonia in pediatric chest radiographs,” *Applied Sciences*, vol. 8, no. 10, p. 1715, 2018.
- [27] D. Kermany, K. Zhang, M. Goldbaum, *et al.*, “Labeled optical coherence tomography (oct) and chest x-ray images for classification,” *Mendeley data*, vol. 2, no. 2, p. 651, 2018.
- [28] “Cc by 4.0.” <https://creativecommons.org/licenses/by/4.0/>.
- [29] “Cc0 1.0.” <https://creativecommons.org/publicdomain/zero/1.0/>.
- [30] R. Leaman, R. Khare, and Z. Lu, “Challenges in clinical natural language processing for automated disorder normalization,” *Journal of biomedical informatics*, vol. 57, pp. 28–37, 2015.
- [31] A. R. Aronson and F.-M. Lang, “An overview of metamap: historical perspective and recent advances,” *Journal of the American Medical Informatics Association*, vol. 17, no. 3, pp. 229–236, 2010.
- [32] P. Mildenberger, M. Eichelberg, and E. Martin, “Introduction to the dicom standard,” *European radiology*, vol. 12, pp. 920–927, 2002.
- [33] L. Budach, M. Feuerpfeil, N. Ihde, A. Nathansen, N. Noack, H. Patzlaff, F. Naumann, and H. Harmouch, “The effects of data quality on machine learning performance,” *arXiv preprint arXiv:2207.14529*, 2022.

- [34] P. Skalski, “Make Sense.” <https://github.com/SkalskiP/make-sense/>, 2019.
- [35] “Gnu general public license,” 2007. <http://www.gnu.org/licenses/gpl.html>.
- [36] “Gnu general public license version 3,” 2007. <https://opensource.org/license/gpl-3-0> [Accessed: 2024/05/22].
- [37] T. pandas development team, “pandas-dev/pandas: Pandas,” Feb. 2020.
- [38] Wes McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.
- [39] “The 3-clause bsd license.” <https://opensource.org/license/bsd-3-clause>.
- [40] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. Software available from tensorflow.org.
- [41] “Apache license,” 2004.
- [42] “Apache license, version 2.0,” 2004. <https://opensource.org/license/apache-2-0> [Accessed: 2024/05/21].
- [43] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.
- [44] J. Cho, K. Lee, E. Shin, G. Choy, and S. Do, “How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?,” *arXiv preprint arXiv:1511.06348*, 2015.
- [45] K. Maharana, S. Mondal, and B. Nemade, “A review: Data pre-processing and data augmentation techniques,” *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, 2022.

- [46] Z. Hussain, F. Gimenez, D. Yi, and D. Rubin, “Differential data augmentation techniques for medical imaging classification tasks,” in *AMIA annual symposium proceedings*, vol. 2017, p. 979, American Medical Informatics Association, 2017.
- [47] N. Garg, B. Agarwal, N. Modi, S. Radhakrishnan, and N. Sinha, “Dextrocardia: an analysis of cardiac structures in 125 patients,” *International journal of cardiology*, vol. 88, no. 2-3, pp. 143–155, 2003.
- [48] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa, “Adam optimization algorithm for wide and deep neural network.,” *Knowl. Eng. Data Sci.*, vol. 2, no. 1, pp. 41–46, 2019.
- [49] “Mit license.” <https://opensource.org/license/mit> [Accessed: 2024/05/28].
- [50] L. Willem, “Pneumonai.” <https://github.com/willeml12/PneumonAI>.
- [51] S. Rajaraman, S. Candemir, I. Kim, G. Thoma, and S. Antani, “Visualization and interpretation of convolutional neural network predictions in detecting pneumonia in pediatric chest radiographs,” *Applied Sciences*, vol. 8, no. 10, p. 1715, 2018.
- [52] S. Candemir, S. Antani, S. Jaeger, R. Browning, and G. R. Thoma, “Lung boundary detection in pediatric chest x-rays,” in *Medical Imaging 2015: PACS and Imaging Informatics: Next Generation and Innovations*, vol. 9418, pp. 165–170, SPIE, 2015.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/epl