Chest X-Ray images classification: Tuberculosis Detector
Applied AI in Biomedicine - Final Project
2022 - 2023

Team members: Alix de Langlais, Felipe Pascutti, Gianmarco Tardini and Pedro Gianjoppe
Team name: Bug Busters
M.Sc. Biomedical Engineering
Politecnico di Milano - Milan, Italy

Abstract - In this project, the objective is to classify chest X-Ray images, which are divided into three classes according to the patient's pathology. Being a classification problem, given an image, the goal is to predict the correct class label. The following report will present the development process chosen to build three different models able to classify the different patients categories according to the input images.

1. Introduction

Today, the most common lung diseases among the population are pneumonia and tuberculosis. Tuberculosis (TB) is caused by a bacterium and is transmitted by air. When a patient has TB, the bacterium multiplies, attacking the lungs as well as other areas of the body. On the other hand, pneumonia is an infection caused by a virus or a bacteria. The alveoli in the lungs are gradually destroyed, making it difficult to oxygenate the body. In order to detect these two diseases, it is possible to carry out X-rays of the lungs. Health professionals will then analyze these images.
Deep-Learning (DL) is a subset of Machine-Learning that enables computers for automatic training and learning of features from a raw dataset. The objective of this project is to implement DL models in order to help in detecting these pathologies in medical images. The main contributions of the paper are listed here:

- Propose and compare three different DL algorithms for tuberculosis detection
- Explainable Artificial Intelligence (XAI) analysis performed over obtained results to provide visualization of the most determining regions of the image
- Use of 'early stopping' in order to overcome the overfitting issue and perform better in real-time

This paper is organized as follows. Section 2 presents the proposed approaches for tuberculosis detection in chest X-Ray images and a description of how the three algorithms were implemented. Section 3 and 4 presents the experimental results and their outcomes. Finally, Section 5 concludes this paper.

2. Materials and Methods

This section of the report will describe the three different approaches implemented for this classification problem (Convolutional Neural Networks (CNN), transfer learning applied to CNN, and machine learning classifier with features extracted with CNN). The dataset, the network architectures as well as the different training strategies will also be described.

2.1. Dataset. The dataset used for this project is composed of 15470 X-Ray images, stored in PNG or JPEG format. These images come from healthy patients, patients with pneumonia or tuberculosis. A .csv file allows the name of a specific file to be associated with the class to which it belongs. The dataset is unbalanced according to the class distribution which is as follows: 9354 images for the normal patients, 4250 for the pneumonia patients, and 1866 for the tuberculosis patients.

2.1.1. Deletion of noisy images. When going through the initial dataset, one can observe that some of the images have spurious noise, making them completely unusable. In a first step, these images were therefore deleted manually, which represented about 5000 items. In order to make the different models to be able to work with noisy images, salt-pepper noise is added to some of the images in the training set at a later stage *(see section 2.1.2)*.

2.1.2. Reduced dataset and sub-folders creation. The dataset provided is composed of more than 15000 images and it has been chosen to reduce it. This allowed to decrease the computation time of the models while keeping very good results. To achieve this, the .csv file containing all the labels associated with file names was browsed. For each class, a fixed number of images is stored in a new .csv file (1200 images per class). This new table contains the names of the selected files and the label of the class to which they are associated. The initial configuration represented an unbalanced distribution of the different classes and this step also allowed to obtain a new balanced dataset. With this new file, a folder for each class is created in order to place the corresponding images.

2.1.3 Split the dataset. Then, the reduced dataset is randomly splitted into training, validation and test sets. For the training phase, 2880 (80%) images are used, for the validation and test phases 360 images are used for each one (10% each).

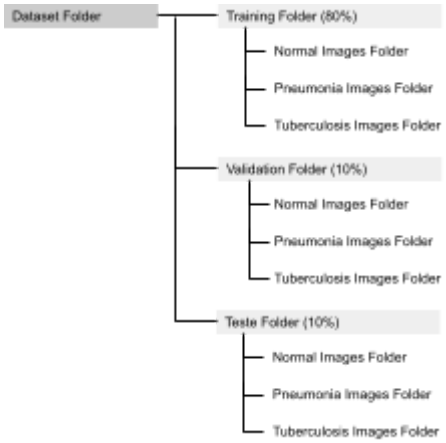

Figure 1. Folders organization

2.1.4. Data pre-processing. X-ray image standards involve a black background and the organ of interest (bone, tissue, etc.) being displayed in white. The proposed dataset has many inverted contrast images. In this way, it was necessary to revert them in order to unify the dataset. By calculating the pixel values in two of the images corners and comparing them with the central pixel values, it was possible to identify the negative images to invert the contrast.

2.1.5. Data augmentation. In order to make the data richer and thus make the model perform better and accurately, some transformations in the dataset were introduced. To do so, five functions were implemented in order to perform the following transformations: rescaling, rotation, shifting, combination of rescale / rotate / shift transformations, and adding noise (salt-pepper noise).

This strategy allows only modifications that are realistics in the X-ray image setting. As the training folder is already composed of 2880 images, it has been decided to apply only one kind of data augmentation on each image. Each image of the original dataset is copied in a new folder. For each of these images, one of the five data augmentation functions is applied randomly and both images, the original and the modified one are saved. At the end of the augmentation procedure, the size of the training set is twice its original one.

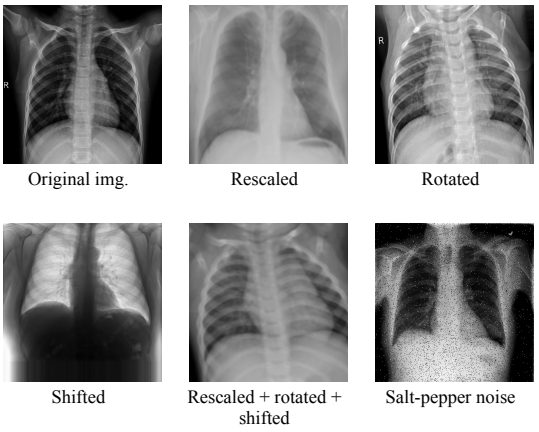

Figure 2. Data Augmentation

As can be seen in the figure, the same image only undergoes one type of transformation.

2.2. Approach 1: Convolutional Neural Network. The Convolutional Neural Network is a subtype of Neural Networks that is mainly used for applications in image recognition. Its built-in convolutional layer reduces the high dimensionality of images without losing its information. For this reason, this kind of model is particularly suited for medical imaging analysis.

The first model presented here is a neural network based on convolutional layers. This type of architecture makes it possible to extract the intrinsic features of an image in order to deduce the class to which it belongs. The following sections will describe the classifier architecture implemented for this first approach.

2.2.1. Max Pooling layer. A Max Pooling layer was implemented to replace fully connected layers in classical CNNs that can be heavy to deal with. It calculates the maximum, or largest, value in each patch of each feature map.

2.2.2. Flattening layer. A flattening layer flatten the pooled feature map into a column vector in order to insert this data into the network later on.

2.2.3. Dropout. A DropOut layer was added in order to prevent overfitting by randomly deactivating neurons.

2.2.4. Dense. Dense layers apply weights to all nodes from the previous layer, so putting them together creates a Multi-Layer Perceptron (MLP) network.

2.2.5. Output layer. One dense layer with softmax activation function was added at the end of the network with three output neurons to match the number of classes.

Layer	Output shape
Input layer	[(None, 256, 256, 3)]
con2d	(None, 256, 256, 32)
max_pooling2d	(None, 128, 128, 32)
conv2d_1	(None, 128, 128, 64)
max_pooling2d_1	(None, 64, 64, 64)
conv2d_2	(None, 64, 64, 128)
max_pooling2d_2	(None, 32, 32, 128)
conv2d_3	(None, 32, 32, 256)
max_pooling2d_3	(None, 16, 16, 256)
conv2d_4	(None, 16, 16, 512)
max_pooling2d_4	(None, 8, 8, 512)
Flatten	(None, 32768)
dropout	(None, 32768)
Dense	(None, 512)
dropout_1	(None, 512)
Dense	(None, 3)

Total params: 18,347,843
Trainable params: 18,347,843
Non-trainable params: 0

2.3. Approach 2: Transfer Learning. Transfer Learning was applied as the second DL approach to take advantage of well-known feature extractor neural network models from literature. It is a method where a model developed for a specific task is reused as the starting point for a model on a second, related, task. The general idea is to use the knowledge one model has learned from a task with a lot of available labeled training data, in a new task that does not have as much data. One of the main advantages of this technique is that it helps in saving time and resources since it decreases the number of trainable parameters.

2.3.1. Transfer Learning models. Well-known feature extractor neural network models related to medical image processing applications were selected from literature. These models already trained on *ImageNet* dataset were loaded, their input shapes were set to (256, 256, 3) and, in order to control the output, their last dense layer was removed by setting the option *include_top* as *False*. Then, for each model, the corresponding pre-processing function was applied to the train, validation and test dataset by means of the function inside the *ImageDataGenerator*() *tfk.applications.[model_name].preprocess_input*.
For the first four models tested (i.e., VGG16, InceptionV3, DenseNet121 and EfficientNetB0), the CNN extractor was followed by a Batch Normalization layer connected to a multi-label Classifier composed by Global Average Pooling, DropOut and SoftMax layers, providing as output the predicted probability vector of the labels.

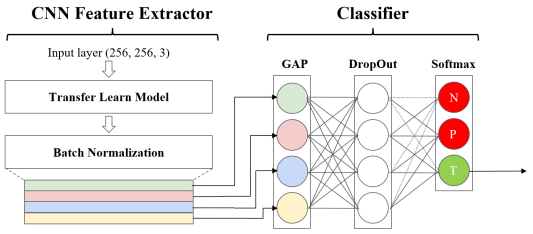

Figure 3. CNN and ML classifier architecture

All the models for Approach 2 were tested in a smaller dataset of 300 samples. This allowed us to compare them in a less time-consuming way. First of all, each model was trained and evaluated with and without the application of the *preprocess_input()* function, and the same was executed for fine tuning.
Then all models were evaluated in the test set by means of classification metrics *(presented in section 2.6)* and the model with best performance was chosen as the effective 2nd approach model. As a last step, it was retrained in the final, larger dataset and its metric was evaluated.

2.3.1.1. VGG16. The VGG16 model has 5 convolutional blocks composed of convolutional layers of 3x3 filter with stride 1 followed by a maxpool layer of 2x2 filter with stride 2. There are 18 layers in total.

2.3.1.2. Inception V3. Inception-v3 has different modules (i.e., 3 Inception A, 5 Inception B and 2 Inception C modules) each composed of several convolutional layers and pooling layers in parallel. Small convolutional layers, such as 3×3, 1×3, 3×1, and 1×1 layers, are used in the Inception modules to reduce the number of parameters [1][2]. All layers summed up in 310.

2.3.1.3. DenseNet121. DenseNet consists of successive DenseBlock layers, each receiving additional inputs from all preceding layers and transition layers, summing up to 426 layers. Additional inputs from all preceding layers together with the feature maps of the current layer are all passed on to subsequent layers [3].

2.3.1.4. EfficientNetB0. EfficientNetB0 is based on the compound scaling technique. It is a combination of depth, width, and input image resolution, these three different dimensions of scaling CNN scale up the width of intermediate features. Additional Flatten, Dense, Dropout, Relu and Sigmoid layers of CNN are structured, summing up to 237 layers.

2.3.1.5. ResNet50. ResNet-50 is a 50-layer convolutional neural network (48 convolutional layers, one MaxPool layer, and one average pool layer). ResNet stands for Residual neural network, a type of artificial neural network that forms networks by stacking residual blocks. At the core of the residual blocks lies the concept of "skip connections" that is the strength of ResNet model.
For the model mentioned [5], the network is then restructured adding a:

- global average pooling layer
- dense layer with 1024 nodes and relu activation function
- dropout layer with 0.7 as rate
- dense layer with 512 nodes and relu activation function
- dropout layer with 0.5 as rate
- last dense layer with 3 units as the number of classes and softmax as activation function

2.3.2. Ensembled Transfer Learning. A last approach tried was the ensemble of the different transfer learning models in order to reduce variance and improve generalization. It was implemented by training each model separately with the same training dataset and then combining their predictions by two different approaches: Major Voting and Averaging.

2.3.3 Fine Tuning. There are two main methods to personalize the pre-trained model. First, is the Functionality extraction, in which the model is used in its entirety and there's a new final classifier that will be trained from scratch. In this way it is possible to reuse the learnt weights from a specific dataset and train the added layer.
Second, is the Fine tuning, in which a chosen amount of superior layers of the model are unfrozen for the specific task with the new dataset. In this way, the unfrozen layers can learn new weights to make the model more consistent with achieving the task. The amount of frozen layers varies with the model and the values were chosen according to the literature.

Model	Frozen layers
VGG16	Until 11th
InceptionV3	Until 228th
DenseNet121	Until 140th

Model	Frozen layers
EfficientNetB0	Until 76th
ResNet50	None

2.4. Approach 3: Machine Learning Classifier. The last approach to be tested consists of two steps. First, a CNN feature extractor which receives as input the X-ray image and performs a data-driven extraction of the most relevant features. Second, a Machine Learning multi-label Classifier that process the input features and provides as output the classification.

2.4.1. CNN Feature Extractor. Two different CNNs were tested to extract the features: VGG19 and CheXNet. Then, the feature vector was collected from either the output of the CNN itself or from an additional dense layer having 512 neurons appended to the CNN followed by Global Average Pooling, to reduce the dimensionality of the feature vector.

2.4.1.1. VGG19. VGG-19 model has 5 blocks, each containing convolution layers with 3x3 filter and an increasing number of channels followed by a max pooling layer [4]. The output shape is 8, 8, 512), which means 512 feature maps with size 8x8.

2.4.1.2. CheXNet. CheXNet model has a 121-layer DenseNet architecture composed of DenseBlocks, each containing convolutional layers with 3×3 filter, ReLU activation and Batch Normalization. The output shape is 8, 8, 1024), i.e., 1024 feature maps with size 8x8.

2.4.2. ML Classifier. The output of the CNN feature extractor was then resaphed as an 1D array to be input of the ML Classifier. Three different types of classifiers were selected to be tested given the intrinsic complexity of the problem. Each one

had its hyperparameters tuned during the training phase and then were evaluated over the test set, being selected as the model with best performance metrics.

2.4.2.1. Support Vector Machine (SVM). The SVM classifier aims to create a linear or non-linear separation hyperplane which splits the data into target classes. The hyperparameters to be tuned are: *C* (regularization), *gamma*, *degree* and *kernel*.

2.4.2.2. Random Forest (RF). The RF classifier consists of training classification trees randomly and then using majority voting to perform classification. The hyperparameters to be tuned are: *n_estimators*, *max_depth*, *min_samples_split*, *max_samples* and *max_features*.

2.4.2.3. AdaBoost (AB). The AB classifier consists of an ensemble learning method that uses an iterative approach to learn from the mistakes of weak classifiers. Then it turns them into strong ones by iteratively increasing the weights for misclassified instances. The hyperparameters to be tuned are: type of weak classifier, *n_estimators* and *learning_rate*.

2.4.3. Hyperparameter search. To perform the tuning of each ML model, the Bayesian hyperparameter optimization search was used. Bayesian statistics is an approach to data analysis based on Bayes' theorem, where available knowledge about parameters in a statistical model is updated with the information in observed data.
The tuning process starts with the definition of the search range for each hyperparameter. Then, the *BayesSearchCV()* function is used to test different combinations of the selected hyperparameters and evaluate them using 10-fold cross-validation. After that, the combination that provided a higher F1-score is selected as the final model.

2.5. Model training All models were trained based on the minimization of the Categorical Cross-Entropy loss function by using the Adam optimizer with learning rate of 1e-4. Moreover, the Early Stopping callback with patience equal to 15 epochs was applied to monitor the validation accuracy in order to avoid overfitting.

2.6. Performance metrics. Different performance metrics were used to evaluate the results of Pneumonia and Tuberculosis diagnostic system by all three models. The most important is the Accuracy (Acc) that identifies the overall correctness of the classification, but supplementary metrics were also needed for a complete multi-class evaluation, such as Precision (Prec), Recall (Rec) and F1-Score (F1).

2.7. XAI. Explainable Artificial Intelligence (XAI) techniques aim to assign understandable meanings to machine learning problems where the parameters involved do not have human understandability. Since in this report medical images are being analyzed, XAI techniques are appropriate and desired.
In general, for image classification, the different methods seek to highlight regions of the image that have greater weight when classifying it. For such, different approaches can be explored. In this project two of the most used techniques were used: Grad-Cam (Gradient-weighted Class Activation Mapping) and Lime (Local Interpretable Model-Agnostic Explanations). [6]

2.7.1. Grad-Cam. Grad-Cam is a technique enabling the investigation of CNNs and the motivations steering neural networks decisions. Grad-Cam method was particularly useful to produce coarse localization maps highlighting the image regions most likely to be referred to by the model when the classification decision is taken.

2.7.2. LIME. Local Interpretable Model-agnostic explanations is a method for training a simple and interpretable model to approximate the decision function of any model such as CNN. To explain a given instance, LIME samples small instances of the image similar to the original one and gets predictions for them using the original model. It then uses the sampled instances and predictions as a new training dataset to which an interpretable linear model is fit. After, the explainer is used to generate explanations in the form of saliency maps depicting regions of an image that contribute to a given prediction.

3. Results

3.1. Approach 1. Given the noise in the initial dataset images, two possible approaches to limit its impact were evaluated. First, a filtering of these images was performed, combining a median filter and a Gaussian filter. Secondly, all noisy images of the dataset were removed and salt-pepper noise was then added in the data augmentation part.
Considering the values on the test set of *Table 1*, it has been decided to keep the dataset with the noisy images added during the augmentation process.

Model	Acc	Prec	Rec	F1
filtering + data aug	0.8806	0.88	0.88	0.88
no noise + data aug	0.9222	0.92	0.92	0.92

Table 1

3.2. Approach 2. For all models, the best test set accuracy was obtained with pre-processing and fine tuning applied at the same time. Therefore these two techniques were used in all subsequent experiments. The following results over the test set from the reduced dataset of 300 samples highlight that the model with the highest test set accuracy is EfficientNetB0.

Model	Acc	Prec	Rec	F1
VGG16	0.8076	0.8974	0.8667	0.8561
InceptionV3	0.8461	0.8593	0.8593	0.8593
DenseNet121	0.8461	0.9394	0.9333	0.9296
EfficientNetB0	0.9333	0.9394	0.9333	0.9296
ResNet50	0.9000	0.9231	0.8899	0.8899
Ensemble	0.8462	0.8472	0.8501	0.8417

Table 2

To compare with the best models from the other two approaches, EfficientNetB0, retrained over the entire dataset provided the following metrics over the test set.

Model	Acc	Prec	Rec	F1
EfficientNetB0	0.9917	0.9913	0.9922	0.9917

Table 3

3.3. Approach 3. First, experiments were performed to evaluate from which layer of the CNN the feature vector should be extracted. Another time the reduced dataset was used to speed up the computation.

Feature vector origin	SVM	RF
Collected from output of VGG19	0.8846	0.6000
Collected from the 512 neurons dense layer	0.7692	0.8846
Collected from the Global Averaging layer	0.8077	0.8333

Table 4

The previous results showed that, for the SVM classifier, the best performance was obtained by collecting the features directly from the output of the CNN feature extractor, while for the RF it was better to collect them from the 512 neurons dense layer appended to the CNN.

Feature extractor	ML classifier	Acc	Prec	Rec	F1
VGG19	SVM	0.9615	0.963	0.9524	0.9548
VGG19	RF	0.8846	0.8804	0.8864	0.8808
VGG19	AB	0.7692	0.775	0.7841	0.7635
DenseNet	SVM	0.8462	0.8889	0.8615	0.8551
DenseNet	RF	0.6923	0.7011	0.6889	0.6899

Table 5

The best performance was then the VGG19 as feature extractor followed by the SVM classifier. Therefore, this model was retrained over the entire dataset.

3.4. Comparison. The results for the metrics parameters (*Section 2.6*) over the test set, were computed for the model of each approach.

Model	Acc	Prec	Rec	F1
CNN	0.9222	0.9200	0.9200	0.9200
EfficientNetB0	0.9917	0.9913	0.9922	0.9917
VGG19 + SVM	0.9900	0.9900	0.9900	0.9900

Table 6

For each model the F1 score over the test set for Normal, Pneumonia and Tuberculosis images was calculated.

Model	F1-N	F1-P	F1-T
CNN	0.8915	0.9108	0.9639
EfficientNetB0	0.9882	0.9959	0.9908
VGG19 + SVM	0.9800	1.0000	0.9800

Table 7

For the EfficientNetB0 model fitting, the following evolution of loss function and the accuracy over epochs for the training and validation set was plotted in order to evaluate the overfitting. As can be seen below, there is some overfitting but according to the good test metrics it does not compromise the generalization capacity of the model.

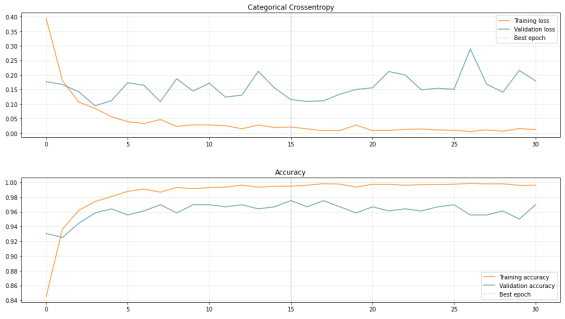

Figure 4. Loss and accuracy evolution over epochs for EfficientNetB0

4. Discussion

4.1. Results analysis. For all three methods used, the metrics are considered good and coherent with the literature. For the first approach, the focus was on investigating how different variations of the

dataset affected the behavior of the neural network. It was found that the dataset that is free of noise images and subjected to data augmentation methods has the best test metrics (*Table 1*).

For the second approach, the main goal was to compare how different existing models performed for the specific case of this project. Firstly, it was investigated which configuration of techniques led to the best result (*Table 2*). Since both pre-processing and fine tuning increase the obtained accuracy, both were kept for the final model. Afterwards, it is necessary to compare how the different models perform in the referred configuration, the best option, chosen for the final model, is EfficientNetB0.

For the third approach, different feature extractors were tested and it was found that VGG19 has the best extraction capacity (*Table 4*). Afterwards, different multi variable classification models were tested in combination with such extractors. The best performance obtained comes from the combination of VGG19 extractor and SVM classifier (*Table 5*).

Finally, a comparison was performed between the three different models based on the metrics obtained, the final model chosen is the best model of the second approach, i.e the transfer learning EfficientNetB0 model followed by a Batch Normalization layer connected to a multi-label Classifier composed of Global Average Pooling (GAP), DropOut and SoftMax layers (*Table 6*).

As shown by *Table 7* it was obtained an F1 score for tuberculosis images greater than 0.75 for all the models, a great result for the reliability of the model.

4.2. Explainable Artificial Intelligence. After analyzing the performance of the models, an XAI analysis to provide explainability for the different approaches was performed. For the first approach, the following techniques were used: Grad-Cam and Lime.

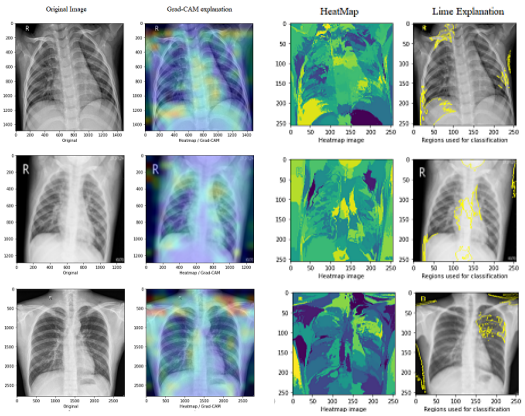

Figure 5. Grad-Cam and LIME for first approach

The same set of images (one for each class) was submitted for XAI analysis of the best model obtained for this approach.

For the images shown (*Figure 5)*, one can notice that, for the Normal class, both reveal an influence of the lower part of the right lung. As for Pneumonia, both indicate an influence of the central part of the left lung, although the other highlighted parts differ between the techniques. Finally, for Tuberculosis, both techniques reveal a strong relation of detection due to the upper part of the left lung of the patient. This analysis will vary according to the patient and the state of the disease.

It is noteworthy that Grad-Cam is a model-based technique, whereas Lime is model-agnostic, that is, it is independent of the trained model. Still, both methods seem to converge for the analysis of the above selected images.

For the second approach, results were obtained by the LIME technique.

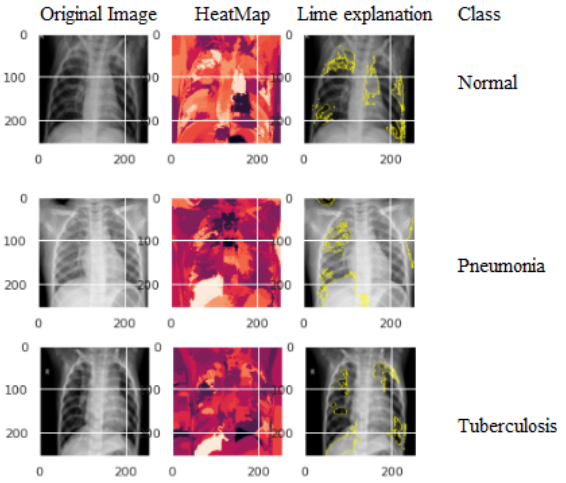

Figure 6. Lime for second approach

Again, one image from each class was subjected to the Lime process. Note that, for the normal class, the most highlighted regions are at the extremities of both the left and right lung. For pneumonia, there is a strong relation of this classification with the right lung. Finally, as regards tuberculosis, one notices that the region responsible for such classification is predominantly present in the left lung of the patient. Once again, this analysis is valid for the presented patients (*Figure 6*) and can vary from one to another.

Finally, for the third approach, it is believed that there is not much sense in applying XAI. Since a ML classifier is being employed, it can be argued that techniques such as Shapley Additive (SHAP) can be used to ascertain which input variables have greater impact for the final classification. However, since in this approach the variables are extracted by

a neural network feature extractor, the inputs passed to the ML no longer have explainability. Thus, although it is possible to say which variables are more relevant, it is not possible to assign a meaning to them.

As it can be observed for some classes in *Figures 5 and 6*, XAI techniques did not always show relevant features for classification, for instance, highlighting the shoulders. It also seems sensitive to model variations and thus, even if the final accuracy is good, DL models do not yet allow full confidence in the results obtained.

Ideally, the explainability tools should converge to point at clinically relevant indicators of Pneumonia and Tuberculosis. If the identified features behind the model predictions are not consistent with medical knowledge about these pathologies, trust cannot be expected from the clinician user. Therefore, a further investigation should be performed together with expert opinion.

5. Conclusion

Image classification problems are recurrent in the medical field. Specifically, in this project, three different techniques to perform the diagnosis of pulmonary diseases were explored.

For each of the approaches mentioned along this report, different methodologies were applied and studied, allowing the integration between the knowledge acquired in the classes and the practical application.

From the acquired knowledge, some topics were investigated aiming to obtain the best performances in the models. Using a balanced dataset, adequate preprocessing phase, and choosing the best hyperparameters of the model, shows to be relevant aspects that were taken into account.

At the end of the experiments, good metrics were obtained in the test set, proving that the models created are satisfactorily implemented. The final model chosen, as already mentioned, is the one that best performs in the second approach (Transfer Learning). That is, the EfficientNetB0 followed by GAP and SoftMax to operate the multivariate classification.

Finally, in order to provide explainability to the models created, XAI techniques were used, within the possible conditions, to investigate the behavior of the different neural networks created and thus evaluate their convergence to clinical relevant indicators of Chest X-Ray images.

6. Bibliography

[1] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. Dans *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. https://doi.org/10.1109/cvpr.2016.308

[2] Guan, Q., Wan, X., Lu, H., Ping, B., Li, D., Wang, L., Zhu, Y., Wang, Y., & Xiang, J. (2019). Deep convolutional neural network Inception-v3 model for differential diagnosing of lymph nodes in cytological images : a pilot study. *Annals of Translational Medicine*, *7*(14), 307. https://doi.org/10.21037/atm.2019.06.29

[3] Fan, R., & Bu, S. (2022a). Transfer-Learning-Based Approach for the Diagnosis of Lung Diseases from Chest X-ray Images. *Entropy*, *24*(3), 313. https://doi.org/10.3390/e24030313

[4] Habib, N., Hasan, M. M., Reza, M. M., & Rahman, M. M. (2020). Ensemble of CheXNet and VGG-19 Feature Extractor with Random Forest Classifier for Pediatric Pneumonia Detection. *SN Computer Science*, *1*(6). https://doi.org/10.1007/s42979-020-00373-y

[5] H. Abdulkader (2021). Covid-19 Diagnosis Using ResNet50 Transfer Learning: Building a DL network from scratch https://www.codeproject.com/Articles/5294463/Restructuring-ResNet50-to-Diagnose-COVID-19

[6] van der Velden, B. H. M., Kuijf, H. J., Gilhuijs, K. G. A., & Viergever, M. A. (2022). Explainable artificial intelligence (XAI) in deep learning-based medical image analysis. *Medical Image Analysis*, 102470. https://doi.org/10.1016/j.media.2022.102470