POLITECNICO
MILANO 1863

ML techniques for X-Ray Chest Image Classification

Author: Vicente Castro

Abstract

This report explores different ML techniques for the classification of chest X-ray images. In specific, it approaches the problem first from a traditional ML perspective, presenting common useful features and classifiers. Later, it discusses established Deep Learning techniques and showcase the benefits of transfer learning against a fully trained network. Finally we discuss

Professor: Valentina Corino

Tutor: Guadalupe Garcia

Course: Applied AI for Biomedicine

September 6, 2023

Contents

1 Introduction

Artificial intelligence (AI) has the potential to greatly enhance the field of medical image recognition, offering numerous benefits to patients, healthcare providers, and researchers. In this project, we have developed a variety of AI models to classify chest X-ray images as 'Tuberculosis', 'Pneumonia', or 'Normal', based on the appearance of the lungs.

This report provides a detailed description of the different approaches used and offers a clear justification for the choices made. It also includes an explainable AI (XAI) analysis of the models to better understand the decision-making process. The results obtained are presented with a clear interpretation, and a comparison is made among the different models. The report concludes with a summary of the findings and recommendations for future research.

Additionally, as the second goal of the project, the comparison between traditional machine learning and deep learning models is presented. Specifically, the difference between Deep Learning and Traditional feature-based machine learning techniques.

2 Material and Methods

2.1 Dataset Description

The dataset comprises 15,470 chest X-ray images obtained from various anonymous patients. These images are accompanied by a `.csv` file containing labels categorising the images into three distinct classes related to validated diseases: *Tuberculosis (T)*, *Pneumonia (P)* or *No-disease(N)*.

Furthermore, it is possible to extract patient information from the image filenames. This allows to assign a unique `patient_id` to each image and, in some instances, discern whether two or more images originate from the same patient. This process yields a total of 12,086 unique patients, with 8,702 of them having only one image and the remaining 3,384 having two images each. It is relevant to note that patients with multiple images all share the same label.

When analysing the labels provided with the `.csv` file, the distribution on Figure 2 is discovered. This distribution remains consistent when considering patients instead of individual images. The pronounced **class imbalance** represents one of the primary challenges in our classification task, as it naturally introduces a bias towards the majority class (N) when training machine learning algorithms.

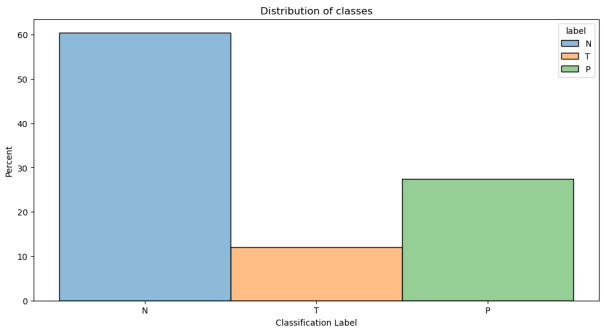


Figure 2: Distribution of samples from the three classes

All the images in the dataset are grayscale representations of the chest area of patients, typically spanning from the shoulders to the lower abdomen. Upon visual inspection it is possible to discover common perturbations affecting the dataset, specifically, three distinct types of perturbations were studied (as presented in Figure 3). Further discussions on these challenges and the proposed solutions for each perturbation can be found in the forthcoming "Image Processing" section of this report.

The first perturbation, the ***added noise***, involves the addition of noise into the images. This noise may originate from diverse distributions, and no single consistent pattern has been identified. The extent of noise varies as well, with some images exhibiting only minor perturbations, while others are nearly unrecognisable (see Figure 4).

The second alteration observed is the ***image inversion***, where certain images are inverted with respect to the traditional X-ray colour scheme. The key challenge posed by this perturbation is the accurate identification of the issue, with the solution being the *bitwise* inversion of the affected image.

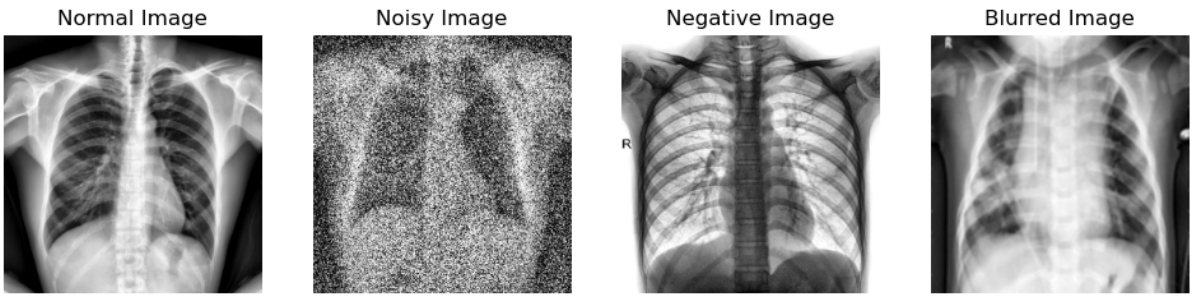


Figure 3: Common perturbation present in the dataset

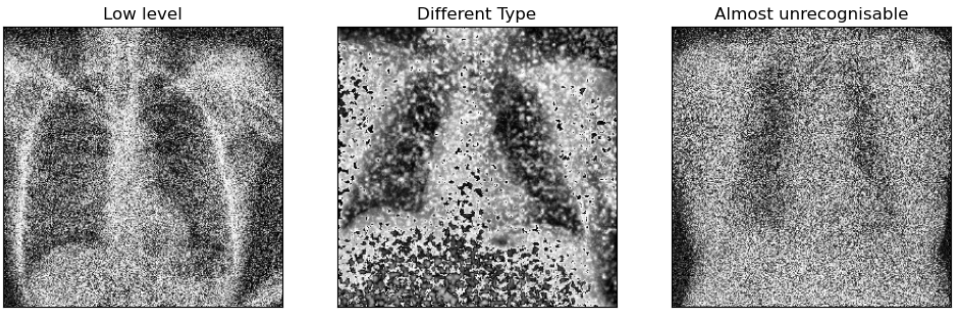


Figure 4: Different types of Noise

The third perturbation is the ***blurring*** presented in some of the images, also applied at different scaled. Different solutions for *sharpening* these images were explored and will be discussed in subsequent sections.

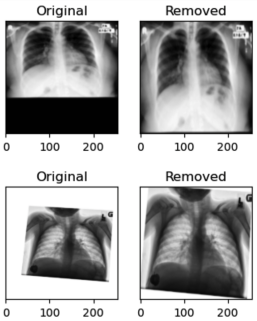
2.2 Image Preprocessing

This section described the preprocessing steps applied to the data in order to normalise it. Each of these presented technique answers to a identified data problem, already discussed or presented in this section. Furthermore, it will be later discussed how this techniques may be more relevant for the *Feature-based models*.

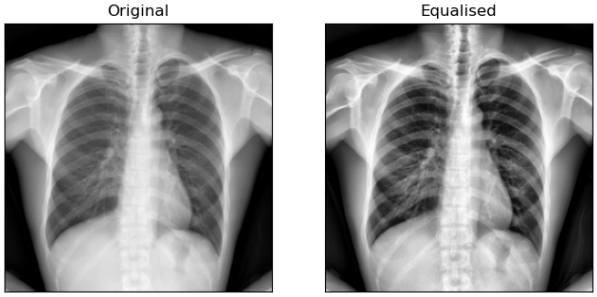
It is relevant to notice that all the hyper-parameters discussed in this section were calculated only from a *training split* taken from the data. The main parameter, the *threshold*, to decide whether an image was affected by any of the perturbation was calculated from the distributions in Figure 10b.

Framed images

The first identified problem comes from a small subset of the dataset presenting "frames" or static bands at the sides (see Figure 5a). The algorithm to clean these images is rather simple: each row and column, starting from each side of the box is checked and removed if its *standard deviation* is less than a threshold. When a row/column does not meet this requirements, the algorithm stops.



(a) Framed images from the dataset



(b) Equalised version of the image

Histogram Equalisation

Histogram Equalisation extends the pixel's intensity range from the original range to 0 to 255, traditionally improving the contrast of the image. This method was applied with a localised version called `CLAHE` (Contrast Limited Adaptive Histogram Equalisation). This alternative is more robust against loss of information due to over-brightness, as it divides the image into small blocks which are then equalised individually. The average result can be observed in Figure 5b.

Noise reduction

Different detection methods for noisy images were analysed to detect in a robust way the noise presented on the image. Finally, the estimator used was based on the *standard deviation* of the pixel values at the corners and at the centre of the image. Gaussian blurring and post sharpening was applied when a certain threshold was met.

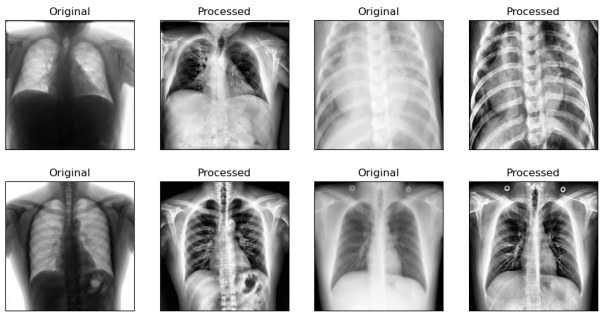
It is relevant to mention that, given the different distributions and scales of noise presented on the images, there was no method that completely succeeded on the removal of noise.

Negative Image Inversion

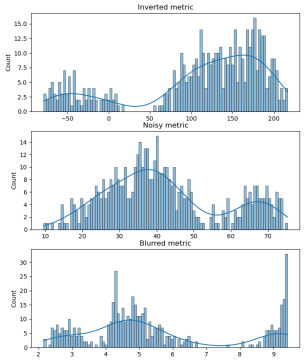
In order to detect negatives, the idea of different section of the image was also used. For this statistic, the mean of the pixel values at the center minus the top corners was used to make a decision. The logic behind this approach is based on what should normally be the position of white pixels on the image.

Image Sharpening

The sharpening of the images was done using a *Laplacian* sharpening. This method consist in computing the second derivative of the image pixels and subtract this values to the original image. The detection was also aided by the different crops, computing the *variance* of the pixels in the *Laplacian*.



(a) Results of the preprocessing pipeline



(b) Threshold for each of the proposed statistics

The final pipeline used for the preprocessing was driven by the performance of utilising in different order each of the discussed steps. Considering this, the taken approach was (1) Cut frames and equalise the image, (2) Detect noise and cancel it, (3) Detect inversion of the image and (4) Sharpen the image. The final results of this method can be seen in Figure 10a.

2.3 Traditional ML

In traditional machine learning, before deep learning era, one of the most relevant steps is the *Feature Engineering* of the problem. Often a process requiring deep understanding of the semantics of the data and the task aiming to solve with it. The general pipeline of feature engineering can be divided in three major steps: (1) Feature Extraction, (2) Feature Selection / Reduction and (3) Model training. In the following paragraph this approach is described.

(1) Feature extraction

As images are spatial related objects, relevant features try to capture this kind of relations. Broadly, in the field of computer vision, three families of features can be defined, based on what on the image activates them.

For each of the following families, several features were studied and discard depending on the how functional they were for the dataset. Special focus must be made to the *texture* features, as is the traditional group with best results in this kind of applications.

1. ***Geometric features***, characterising the shape of the bodies on the image. Among the features analysed are:
   (a) The four Flusser Moments (*discarded*)
   (b) The seven *Hu Moments*, applied on the binary enclosed image.
   (c) Fourier Descriptors, taken from the Fourier transform of the image, 32 descriptors were kept.
2. ***Pixel Intensity***, characterising the distribution of the pixel values on the image.
   (a) Statistical Features: Mean, standard deviation, kurtosis, skewness
   (b) Gabor Features, calculated from applying different convolution of Gabor-filters with the image.
3. ***Textures***, probably the most widely used and rich family of features. They are based on complex local-structures associated with the gradients or pixel patterns.
   (a) Local Binary Patterns (LBP)
   (b) Histogram of Oriented Gradients (HOG) (*discarded*)
   (c) Haralick Features, measuring correlation values on the image.

Most of the hyper-parameters for this set of features were studied based on how much *variance* and *separability* they carried. This is a natural approach as, in the context of X-ray images, the dataset may seem too similar under features (the i.e: silhouettes of the chest are similar in most humans).

Furthermore, one of the most traditionally used features, *Histogram of Gradients*, proved not to be fitted for this dataset. This can be explained by the amount of noise present in the images and how this feature reacts to it. In Figure 7 both the original image and the calculated histograms can be seen, this provides a good visual example of how the feature is not able to compute meaningful values in presence of noise.

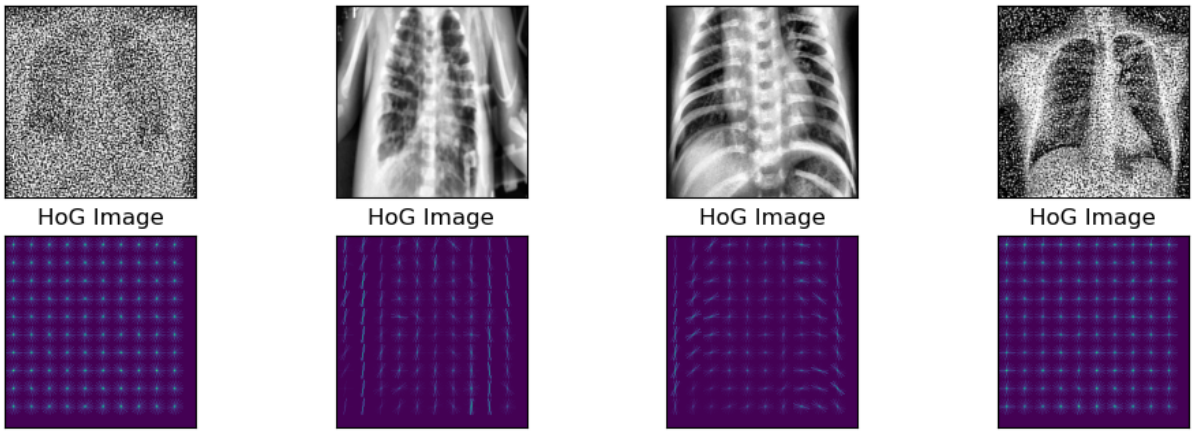


Figure 7: HoG calculations on sample images

The final number of features, under these assumptions is 247 per image.

(2) Feature selection

Feature selection procedures are based on the idea of obtaining the best set of minimal features for the dataset. Among different techniques to accomplish this a difference can be made between iterative processes (i.e: *Backward Feature Elimination*), based on removing/adding the best next candidate feature given a targeted classifier, and statistical approaches, where the reduction is made without a classifier and trying to keep most of the information carried by the features.

The analysis was centred around the last idea, how to keep the major part of the information, while removing less relevant features. For this, two ain approaches were used: **Variance Thresholding** (VT) and **Principal Component Analysis** (PCA).

VT drops features whose variance falls below a specified threshold, while PCA reduces the dimensionality of a large dataset by projecting it onto a smaller set of principle components that explain the variance in the data. Later the kept or projected features can be used to train the classifier. In both cases, the selection of the final number of features was based on maintaining a minimal amount of explained variance in the final features sets.

Finally, to maintain some posterior explainability on the features, the reduction technique was applied to the features as grouped by the discussed families. This procedure can be seen in Figure 8, where reducing the geometrical features from 132 to 96 is the optimal reduction according to the expected explained variance.

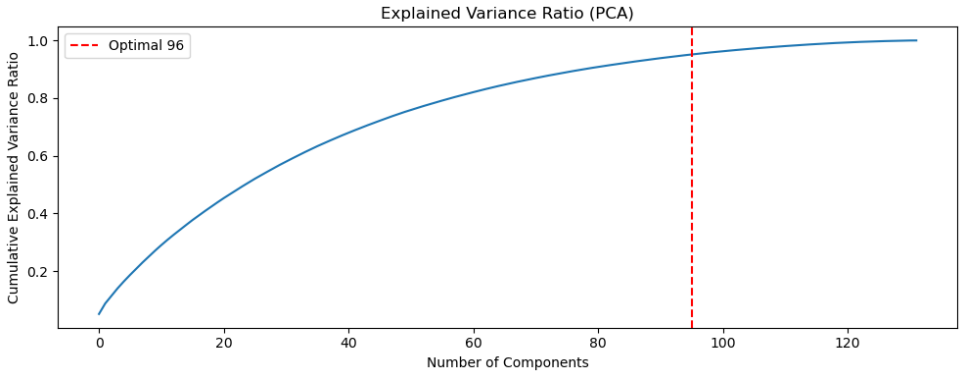


Figure 8: Optimal number of components for Geometry family

(3) Modelling

For the classification of the features into their corresponding classes, two models were analysed. These were mostly used due to their broad validation in this kind of application, in fact, both were kindly regarded as the *state-of-the-art* before Deep Learning came into the scene.

Support Vector Machine (SVMs), are a type of supervised machine learning model that separates data points into different classes by finding the optimal hyperplane in an N-dimensional space. The SVM aims to maximise the margin between the classes. When the data is not linearly separable, a kernel function can be used to map the data points to a higher dimensional space where the points can be separated by a hyperplane. Common types of kernel functions include linear, polynomial, Radial Basis Function (RBF), and sigmoid.

Random Forest Classifier, is a kind of learning algorithm based on the ensemble of several "weaker" models. In the case of Random Forest, the weak models are *Decision Tree*, each one being trained with random splits of the train data. The final classification for a sample is made by aggregating the classification on all the trees, usually by majority voting. The more trees a forest has, the more the capacity to fit the data it has and the more robust it should be (under good treatment of the data).

Both of these models have different parameters that need to be tuned. This step is crucial and is discussed in the Results section.

2.4 Deep Learning

Deep Learning (DL) as a learning technique, changed vastly the traditional machine learning paradigm. The crucial difference is in the *Feature Engineering* process, with DL the discovery of meaningful features is done in a *data-driven* way. This means, the algorithm itself learns which features are more relevant for the task.

For images, the most used model are *Convolutional Neural Networks (CNN)*, a type of neural network based on local-connectivity and parameter sharing. This two concepts, based on the the actual mechanism of vision neurons, make these family of model particularly well-suited for image recognition tasks.

These networks are designed to process data through a series of layers that learn to identify patterns in images. The layers in a CNN typically include convolutional layers, pooling layers, and fully connected layers.

Convolutional layers, are used to identify features, such as edges, corners, and textures. These layers use a mathematical operation called *convolution* to scan the image and extract the meaningful values.

It is relevant to mention that, depending on how deep is the layer, the features extracted are usually an aggregation, reaching the point of a layers recognising meaningful structured.

The ***pooling layers*** are used to reduce the dimensionality of the data by down-sampling the feature maps. This helps to reduce computational cost and prevent overfitting.

Finally, ***fully connected layers*** are used to classify the image based on the features that have been identified in the previous layers. These layers connect every neuron in one layer to every neuron in the next layer. The final layer of the network typically carries a softmax operation to output a probability distribution over the different classes.

In this analysis, three different CNN architectures are studied, with additional techniques as transfer learning and fine-tuning. Methods that are able to port the general features learnt on different datasets to achieve better results. The following models were used:

Hand-written CNN model

This model is a smaller, hand-written architecture composed by only Conv2D layers, BatchNormalization layers and MaxPooling layers. This model is trained from scratch and does not utilise pre-trained weights. It is designed with the purpose of being simple yet efficient.

The classification head present on the model, takes the features extracted by the convolutional layers and, using a FF Neural Network, classifies into one of the three classes. In particular, this head is composed by:

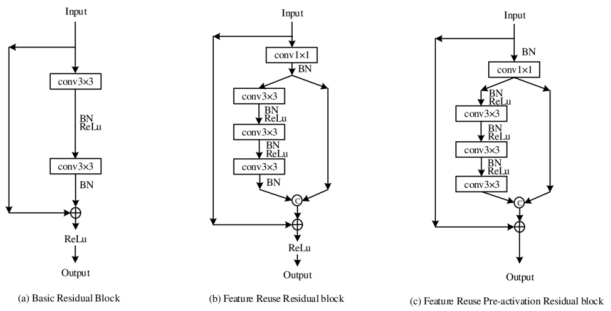
- `Dense layers`
- `Dropout layers:` The Dropout layer randomly sets input units to 0 during training time to help prevent overfitting.
- `Batch normalisation:` is a technique used to improve the stability and speed of training by normalising the layers inputs.

ResNet

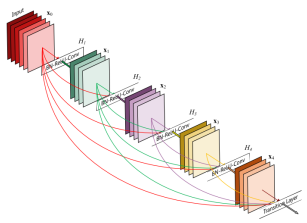
This model is built using the ResNet-50 architecture, which is a 50-layer convolutional neural network that uses residual blocks to form the network. The ability of this model to generalise comes from the gradient passing mechanism allowed by the residual blocks. In practice, this mean that the model is able to maintain more stable gradients during the training phase, even on early layers. This model was used to perform *transfer learning* for our task, so the weights trained on *ImageNet* dataset were used.

DenseNet

This model is built using the DenseNet-121 architecture, which is known for its ability to make deep learning networks more efficient by using shorter connections between layers. Similarly to the ResNet model, only the backbone was used, after which a classification head was added. The weights from *ImageNet* were also used.



(a) Residual block from ResNet



(b) DenseLayer from DenseNet

2.5 Explainable AI techniques

In this section, the two methods for *Explainable AI* used in the anylisis are presented.

2.5.1 Shapley Additive Explanations (SHAP)

Shapley Additive explanations (SHAP) is a model agnostic approach to explain the output of a machine learning model that originates from game theory. SHAP computes the importance of a feature by seeing how well the model performs with and without that feature for every combination of features, by computing shapley values for each feature value. The shapley value is the average marginal contribution of a feature value across all the possible combinations of features.

This technique was used to evaluate the relative importance of features in the Random Forest model.

2.5.2 XAI with GradCAMs

For the deep learning approach, GradCAMs (Gradient-weighted Class Activation Mapping) of the output class were computed. This technique is a method for visualising which regions in an image are most important for the prediction made by a CNN model. It is a type of explainable artificial intelligence (XAI) method that can be used to understand the decision-making process of a CNN, and can help to identify the specific features in an image that the network is using to make its predictions.

The method works by computing the gradient of the class score with respect to the feature maps in the final convolutional layer of the CNN. These gradients are then averaged across the different feature maps and are used to weight the feature maps, effectively highlighting the regions in the image that have the strongest influence on the class prediction.

The importance of this technique lies in its ability to provide a visual explanation of the decision-making process of a CNN, which can be particularly useful when working with medical images, where the interpretability of the model is crucial. Additionally, it can be used to identify the regions in the image that are most relevant for a given class prediction, which can be useful for further data analysis and model improvement.

3 Results

To test the proposed models two splits, at different level, of the dataset were done. Firstly, the dataset was divided into a `training` and `testing` sets, setting the development and parameter tunning of the models only with the training split. With this split all the pre-processing hyper-parameters were calculated. The percentage of the dataset kept for testing purposes was 20%.

Secondly, and for a more robust model selection, *cross-validation* was used to create different validation scenarios from the training data. This method is vastly used and allows to have less biased statistics for the performance of the models as opposed to only have one *validation* set extracted from the training data.

Furthermore, for both splits, the testing and the cross-validation, a stratified sampling was used based on the unique patients and the labels. This is crucial to avoid any data leakage from the training to the testing settings and to maintain a similar distribution of classes among the different splits.

3.1 Features-based models

For each of the models, a grid-search was performed to obtain the best set of parameters and hyper-parameters. This second set of setting consisted on what preprocessing to do, and what kind of feature reduction technique to used.

Particularly, for the preprocessing, two image sizes were used 224 and 128 pixels per side. Additionally, each of the preprocessing steps were tested, trying to understand which techniques were more relevant for the models.

The following are the grids used on the fine-tuning of each model. The `DATA_SUBSET` are all the different combinations of image size and preprocessing steps applied.

```python
# gridSearch for SVM
param_grid = {
    'C': [1, 5, 10],
    'gamma': ['scale', 'auto'],
    'kernel': ['poly', 'rbf'],
    'subset': DATA_SUBSETS
}
```

(a) GridSearch on SVM

```python
# gridSearch for RandomForest
param_grid = {
    'n_estimators': [50, 100, 150]
    'subset': DATA_SUBSETS
}
```

(b) GridSearch on RandoForest

Support Vector Machine

Based on the grid the following results were obtained:

C	Gamma	Kernel	Subset	Ft_type	acc	f1	f1_macro
10	scale	rbf	small_with_noise_inverted	raw	0.797704	0.828090	0.723493
5	scale	rbf	small_with_noise_inverted	raw	0.797704	0.828090	0.723493
10	scale	rbf	medium_with_noise_inverted	raw	0.791809	0.826212	0.718469
5	scale	rbf	small_with_noise_inverted	raw	0.783742	0.823188	0.712252
10	scale	rbf	medium_without_noise_not_inverted	raw	0.771570	0.813256	0.695784

Table 1: SVM results, top 6

The results present some interesting insights, the major one being that the model is able to learn fine from the *raw data* (without any feature reduction technique). Furthermore, it shows preferences towards applying the *Image Inversion* detection, but without noise reduction.

Note that the parameters for *gamma*, "scale" and "auto" are an scaling based on the variance of the features. In this case, for the pipeline of "raw" the values were standardised, so the value is the same

When applying, both reduction techniques the performances were lower, specially in the case of *PCA*, being at the bottom of performance. The best PCA and VT results are presented in Table 2:

Random Forest

The Random Forest classifier results are presented in Table 3. The results are slightly better than the ones obtained for the *SVM*. Furthermore it is interesting to check that the values are better for bigger

C	Gamma	Kernel	Subset	Ft_type	acc	f1	f1_macro
5	scale	rbf	small_with_noise_inverted	raw	0.689634	0.717241	0.607571
1	scale	rbf	small_with_noise_inverted	raw	0.555556	0.598158	0.489180

Table 2: SVM best results for PCA and VT

images and without preprocessing techniques applied to them. This could be a good indicated of the model being able to extract deeper meaning from the images as a sort of "data augmentation".

The results for PCA and VT techniques also yielded lower performance values.

N_Estimators	Ft_type	acc	f1	f1_macro	Subset
150	raw	0.807406	0.848005	0.615830	medium_without_noise_not_inverted
100	raw	0.807199	0.846912	0.624591	medium_without_noise_not_inverted
50	raw	0.805130	0.842042	0.627265	medium_without_noise_not_inverted
150	raw	0.810923	0.831419	0.645681	medium_with_noise_inverted
100	raw	0.808233	0.829220	0.638075	medium_with_noise_inverted
150	raw	0.796649	0.825074	0.605514	small_without_noise_not_inverted
100	raw	0.796649	0.822028	0.608838	small_without_noise_not_inverted
50	raw	0.807199	0.821037	0.641650	medium_with_noise_inverted

Table 3: Rando Forest results

In summary, the Random Forest model performed better than the SVM in classifying the X-ray images. The Random Forest model trained on all extracted features performed and without preprocessing performed the best, while the best SVM model was helped by the preprocessing steps.

3.2 Deep Learning models

As in the previous experiments, a grid-search with cross-validation for the hyperparameters tuning of the model was perfored. The DenseNet and ResNet models were fine-tuned using pre-trained weights from `imagenet`, while the Simple CNN was trained from scratch. The metrics to evaluate the performance of the models were F1-Score and Accuracy. The most important metric was the F1-Score obtained for the Tuberculosis class.

For the DenseNet and ResNet models, different combinations were tested for the number of layers in the classification head and the number of layers unfrozen for the fine-tuning. For the simple CNN, different numbers of CNN layers were included.

As the analysis was performed ore on a *structural* architecture, the learning rate and weight decay were set at the default values, $1e^{-3}$ and 0, respectively. To avoid overfitting a callback for *EarlyStop* was used.

The following are the three best results obtained for the ResNet and DenseNet based models:

F1_N	F1_P	F1_T	Acc_N	Acc_P	Acc_T	Head L	Unfrozen L	Model
0.95	0.97	0.85	0.94	0.98	0.89	2	10	DenseNet121
0.95	0.97	0.84	0.94	0.98	0.87	3	10	DenseNet121
0.95	0.97	0.84	0.94	0.98	0.86	3	5	DenseNet121
0.96	0.97	0.83	0.96	0.98	0.83	2	10	ResNet50
0.95	0.97	0.83	0.95	0.98	0.84	4	10	ResNet50
0.95	0.97	0.83	0.95	0.98	0.82	3	10	ResNet50

And the following are the best results for the SimpleCNN model:

F1_N	F1_P	F1_T	Acc_N	Acc_P	Acc_T	Layers	Model
0.90	0.94	0.70	0.87	0.96	0.77	3	SimpleCNN
0.90	0.92	0.66	0.88	0.93	0.74	7	SimpleCNN
0.88	0.92	0.63	0.82	0.93	0.83	5	SimpleCNN

From the results, it can be seen that the DenseNet model with 2 layers in the classification head and 10 layers unfrozen for fine-tuning had the best F1-Score for the Tuberculosis class, with a value of 0.85. The ResNet model with 4 layers in the classification head and 10 layers unfrozen for fine-tuning also had a

high F1-Score for the Tuberculosis class, with a value of 0.83. The best results for the simple CNN model were obtained with 3 CNN layers, with a F1-Score of 0.70.

These results are in line with what was expected from the fine-tuned models, as they were trained using a pre-trained weights and therefore, have a better feature extraction capability. Also, the fact that DenseNet and ResNet have more layers than the simple CNN, increases the representational power of the models, and that is why they performed better.

Furthermore, the best results were obtained with a relatively low number of layers in the classification head and a relatively high number of layers unfrozen for fine-tuning. This is because having a large number of layers in the classification head may lead to overfitting, while unfreezing more layers allows the model to adapt better to the specific task of tuberculosis classification.

Performance on Pre-processed Images

When testing the two best models on the preprocessed images (same process as described earlier), the performance didn't increase, but it was relatively the same. For the best DenseNet model, the F1-Score for tuberculosis, had only a difference of a 0.001 less than when trained in raw data.

This can be explained by the fact that models such as ResNet and DenseNet, are able to learn complex features from the raw data by themselves, and the need for any preprocessing technique may be reduced compared to traditional methods. In other words, they also learn a "internal" preprocessing when they are extracting the features, therefore, this early steps may be skipped.

3.3 XAI analysis

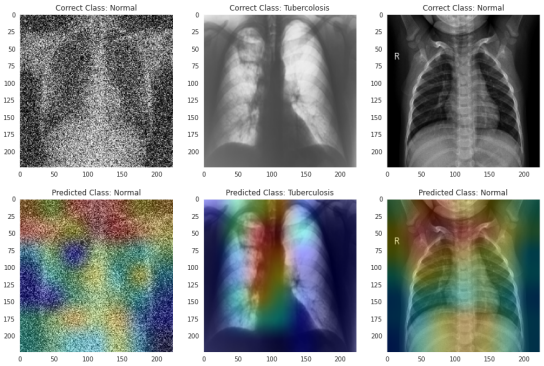
Shapley Additive Explanations

The SHAP analysis was applied to the best RandomForest model from the grid search.

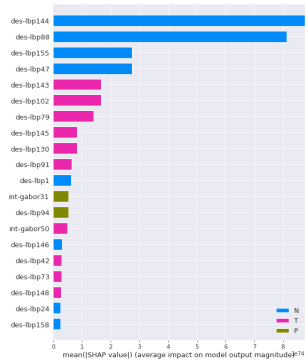
To compute the SHAP values, the "fast" SHAP function was used, which uses a sampling-based approximation algorithm. This method is typically faster than other methods for computing SHAP values. The bar plot in Figure 11b presents a summary of the mean SHAP value for each feature among the top 20 most important features. The plot demonstrates that different LBP descriptors are the key features for the model being evaluated.

XAI with GradCAMs

The Grad-CAM visualisation for the tuberculosis class indicates that the model is effectively identifying regions of the image that contain anomalous formations, which is consistent with the high performance metrics observed for this class. For the normal class, the model appears to be focusing on more general regions outside of the lungs, even in the presence of noise. This may indicate that the model is robust to the presence of noise, however it may also suggest overfitting to certain features present in the training set.



(a) GradCAMs on the three classes



(b) SHAP on Random Forest

4 Discussion and Conclusion

One of the main challenges in the provided dataset is the class imbalance. This problem makes the models more prone to overfitting to the majority class, as they see more examples of it during training. Even when using techniques as class weights for the training of our models, the overall decrease in performance is notorious in most of the cases. For future iterations, a good idea to tackle this problem could be generating some synthetic samples for the minority class.

Feature-based methods

In terms of preprocessing, it is important to note that these techniques can be used to enhance the features that are relevant to the classification task. For example, histogram equalization can be used to enhance the contrast of an image, making it easier for the model to identify important features such as edges and textures. This step is especially important when we are extracting features based on the pixel intensity or the texture in the images. In the future, it would be interesting to compare the performance of these models on the raw images and on the intermediate steps of the preprocessing, to check the impact each step has.

In terms of the results, it is possible that the SVM classifier performed better than the Random Forest classifier because SVM is a linear model, and therefore it can be more robust to high-dimensional data. Additionally, SVMs have the ability to handle non-linearly separable data through the use of kernel functions. The Random Forest Classifier, on the other hand, is an ensemble method that can be sensitive to the quality of the data and the number of samples, which can lead to overfitting if not used carefully. Nevertheless, when inspecting the performances of the other classes, is clear that with the right amount of data improvements are possible.

Our results are consistent with the literature in X-ray image classification, where SVM has been frequently mentioned as a promising approach. Additionally, these studies also indicate that Random Forest typically performs worse for this type of image classification.

Lastly, regarding the poor performance of PCA as a dimensionality reduction technique, the primary reason for this behavior can be attributed to low correlation between features, making it difficult to identify PCA components that describe the variance between them.

Deep Learning Models

The CNN models we used in this study, DenseNet and ResNet, are well-used architectures for image classification tasks. As expected, the results obtained with these models were superior to those obtained with the feature-based methods and the vanilla CNN model. The best results were achieved with the DenseNet-CNN model, which got to an average F1-score of 0.92. These results demonstrate the power of CNNs in capturing the complex patterns present in chest X-ray images.

However, despite the good performance of these models, there are some limitations that need to be addressed. One of the main concerns when using DL models is their lack of interpretability. While the GradCAM visualization provides some insight into the regions of the image the model is focusing on, a more comprehensive interpretability method could be explored.

Finally, to improve the performance of the models, some potential future improvements include incorporating more data augmentation techniques, using more complex fine-tuning methods such as Bayesian optimisation, and experimenting with different architectures like Inception and EfficientNet. Additionally, it would be interesting to evaluate the performance of these models on larger datasets and different types of chest X-ray images, such as high-resolution images or images from other modalities such as CT scans.