UNIVERSITY OF OSLO

Project I

Applied Data Analysis and Machine Learning

Involved students

Johan Blakkisrud
Toralf Husevåg
Frederik Eichenberger

Professor: Morten Hjorth-Jensen

Oslo, December 18, 2023

1 | Abstract

Pneumonia stands as a global health concern, often fatal and exacerbated by healthcare disparities. This study explores an innovative approach for X-ray image classification of pneumonia. Integrating handcrafted features into conventional machine learning models, including feed-forward neural networks (FFNN) and others, offers a comparative analysis against CNNs, aiming to elevate pneumonia detection on a dataset from the RSNA detection challenge of 2018 methodologies. Radiomic feature based models achieved accuracies slightly above 80% while the CNN model was slightly below.

The project code can be found at https://github.com/toralfhus/MachineLearningProject3/.

2 | Introduction

The severity of pneumonia, coupled with varying co-morbidities and disparities in available healthcare infrastructure, renders it a potentially fatal disease. Globally, millions are affected, with approximately 740,180 reported child deaths in 2019 alone (WHO, 2019). Particularly in underprivileged regions, pneumonia poses a significant concern due to the lack of medical resources.
The gold standard for early pneumonia detection involves high-cost imaging modalities like computed tomography (CT) and magnetic resonance imaging (MRI), where CT is considered superior [1]. In contrast, X-ray serves as a relatively low-cost alternative suitable for large-scale implementation even in resource-limited settings. Despite its lower resolution compared to CT, X-ray remains a primary initial diagnostic tool, especially where advanced techniques may not be readily available.
Given the scarcity of radiologists and the routine use of X-rays, automated pneumonia detection has garnered substantial attention [2]. Deep learning, particularly convolutional neural networks (CNNs), has emerged as a highly successful strategy for image classification problems.
In parallel, radiomics, an evolving field in radiology, integrates textural and shape-derived image features with machine learning, unveiling insights beyond visual inspection. This approach has witnessed extensive exploration within computational radiology.
This study aims to implement radiomics as an alternative approach for classifying X-ray images by extracting handcrafted features and coupling them with conventional machine learning, including feed-forward neural networks (FFNN), random forests, logistic regression and XGBoosts. This methodology will be compared with a simple CNN implementation and critically analyzed against state-of-the-art findings in the literature.

3 | Theory

3.1 | Radiomics

Used as an umbrella term for applying machine-learning based statistical inference methods to medical imaging, *radiomics* attempts to utilize quantitative information found in the images to predict clinical outcomes. The general hypothesis is that using machine learning may utilize additional information invisible to the naked eye, even for a trained radiologist, and may as such be used as a decision-helping tool in the clinical setting. The term is usually used when applying pre-defined mathematical procedures to extract (compute, mine) the imaging features, known as *hand-crafted* radiomic features, but the use of deep learning "directly" on the image (such as a convolutional neural network, see chapter about CNNs) is referred to as *deep radiomics* by some authors. Combining the two modalities of radioimcs, i.e. handcrafted and deep radiomics, is an area of active research [3].
The imaging biomarker standization initiative (IBSI) attemts to standardize both the mathematical operations used, and the pipeline, for extracting hand-crafted radiomic features [4]. Said pipeline is illustrated in figure 3.1 and includes segmentation (see section 3.3.1), re-segmentaiton (exclusion of voxel outliers), normalization, and gray-level discretization.

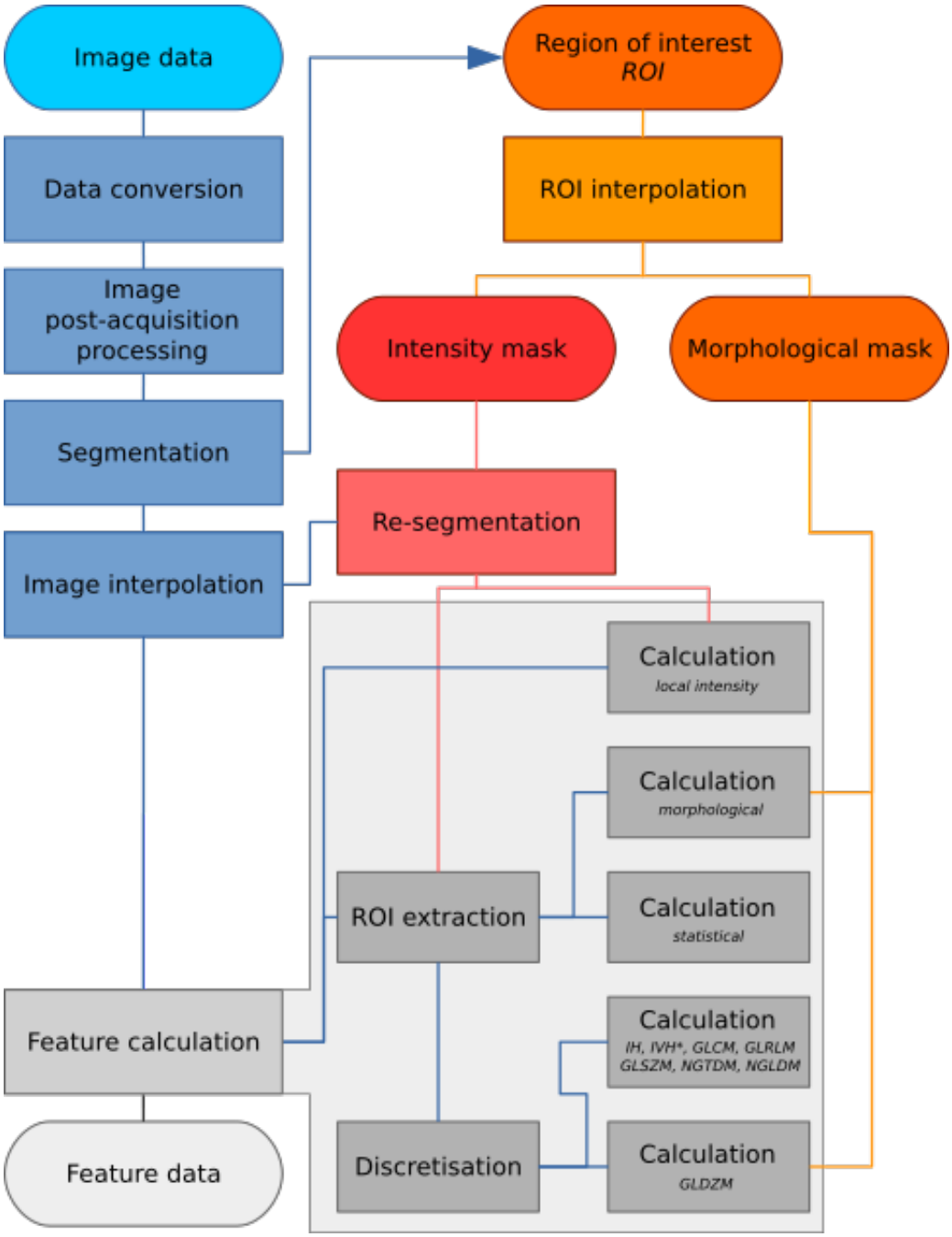

Figure 3.1: The image biomarker standardization pipeline for extracting hand-crafted radiomic features [4].

The hand-crafted radiomic features are generally divided into three categories: shape-based and histogram-based (first-order) radiomic features, and texture-based features. First-order or textural features may be calculated without or with a filter applied beforehand, e.g. a gaussian smoothing or wavelet transform. To compute the textural features a intermediate matrix is caluclated, such as the gray-level co-occurance matrix (GLCM; see figure 3.2) counting the number of times two voxels are neighbours (co-occurs) along some direction. The size of said matrix is dependent of the number of individual gray-levels in the image, and therefore discretization of gray-levels is performed before the calculation of the matrices. Whether to use a fixed bin width with variable bin count (FBW) for discretization, or vice versa (fixed bin count; FBC), have not reached a scientific consensus but is an area of research to find the appropriate balance between noise suppression and signal truncation. However, some studies using 18F-FDG Positron Emission Tomography (PET) based radiomic features to predict treatment response to radiotherapy of lung cancer and radiochemotherapy of espohageal cancer used between 32 and 128 bins (for FBC), or a bin width

resulting in a similar amount of bin counts (for FBW) [5][6] .

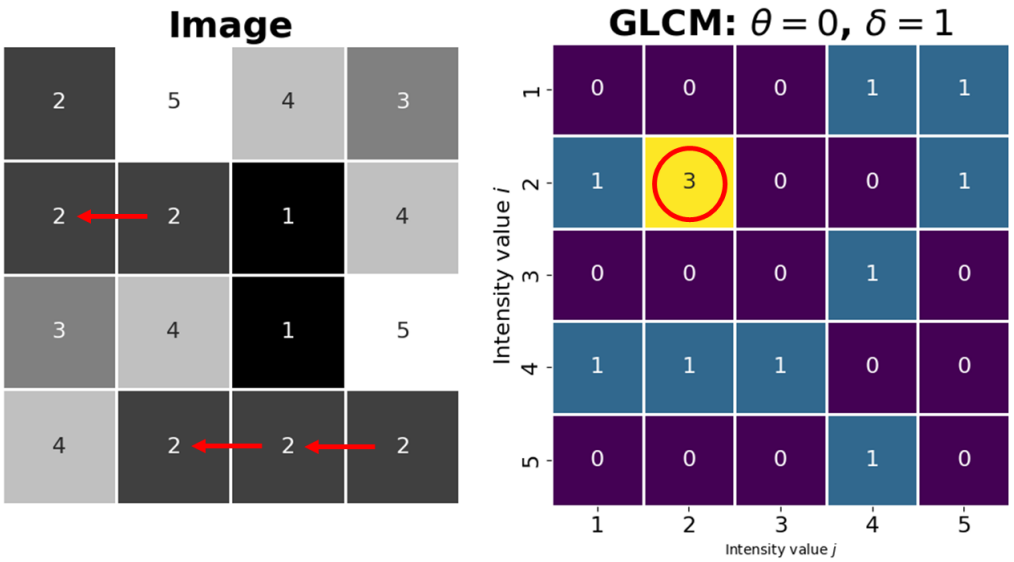

Figure 3.2: Calculating the GLCM for a discretized image of $N_g = 5$ gray levels (left), resulting in a 5 × 5 GLCM (right). Having 3 instances of gray level $i = 2$ neighbouring the same value ($j = 2$) at distance $\delta = 1$ in the right horizontal direction $\theta = 0$, yields entry (i, j) = 3 in the GLCM. Image created by the author using the python package pyRadiomics [7].

3.1.1 | Feature selection in high-dimensional radiomics

Real-world data is usually not uniformly distributed, residing in a lower effective dimension than the measured data (i.e. the intrinsic dimensionality) [8]. While IBSI considers the extraction of hand-crafted radiomic features, the machine learning methods utilized post-extraction is outside the scope of the initiative. Due to the high-throughput nature of radiomics (i.e. the extraction of a high number of features per image), *feature selection* is commonly applied to reduce feature redundancy and complexity in the final model. This procedure should be considered a part of the *learning phase* for a machine-learning pipeline to avoid overfitting the reduced feature set to all of the data (i.e. data leakage), and should therefore be performed after reserving some of the data for testing the generalized model performance [9]. The Least Absolute Shrinkage and Selection Operator (LASSO), or L1-norm, is an effective and commonly used method for feature selection by regularization as it allows some of the model coefficients to become exactly zero (i.e. discarded). Another commonly used method is to recursively train a model and remove the features with lowest importance in the model (e.g. weights), until the appropriate number of features is reached (e.g. by minimizing cost on a validation set), known as recursive feature elimination (RFE) [10].

3.2 | Feed Forward Neural Networks

In a regular feed-forward neural network (FFNN) each neuron receives an input from each neuron from the previous layer or the initial input. These are also called fully connected, or dense layers. Each neuron has its own weight and bias influencing the values it receives. After adding the weight and bias to a value it is passed through an activation function, which introduces non-linearity into the model. For improving a model one can test out different activation functions, but most commonly used is the Relu function, which is defined as

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0. \end{cases} \tag{3.1}$$

After the last layer a different function is usually employed to get the output. In case of a binary classification problem most commonly the sigmoid function is used which is given by

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{3.2}$$

To start things off the weights and biases are initialized using random numbers and the error of the prediction is calculated and compared to the actual values, which is done by a loss function. For binary classification the Cross-Entropy function is the most common one, defined as , defined as

$$\mathcal{C}(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \left(y_i(\beta_0 + \beta_1 x_i) - \log\left(1 + \exp\left(\beta_0 + \beta_1 x_i\right)\right)\right). \tag{3.3}$$

Depending on the error the weights and biases will be adjusted in a backwards algorithm, which relies on using the differential chain rule to calculate each layer iterative with respect to the loss function.
Often the loss function is also regularized, using L2-regularization which in terms prevents the weights from obtaining too large values and aims for more balanced weights with increasing importance of the regularization term, which can be described by the pre-factor or regularization parameter λ.

3.3 | Convolutional neural network

CNNs are a class of feed-forward neural network that learns features using optimization by filters. After a filter is applied onto the input data the output is usually called activation map. These take far less time to compute compared to regular FFNN, and take up less memory. Thus, convolutional neural networks are a powerfull tool especially cases in which the input data is in the form of images.
The architecture follows the basic structure of other FFNNs, with some notable differences, which are found in the layer structure, since usually different layers are used, than the earlier mentioned dense layer.

Convolution layer: In this layer we apply a given amount of filters onto the input data, thus returning activation maps for each applied filter. A neuron takes up only information from the local receptive field, which is covered by the size of the filter. This is also referred to as sparse interaction, since the neurons are now calculated from less values. This reduces the memory needed as well as the computational time since fewer weights need to be calculated. Furthermore, since the filter with the same weights is applied onto the entire input or previous layer it is possible to learn local patterns while also reducing the amount of memory needed [11]. The filter is shifted over the entire input, or previous data. So e.g. if the previous data was of the dimension 40 x 40 and a filter of 5 x 5 is used, we obtain an activation map of the dimensions of 36 x 36, if a stride length of 1 is used. The stride length defines by how much the filter is shifted each time, which can be varied in testing CNNs. To not loose the given dimensionality one can also introduce a zero-boundary by adding a frame of zeros around the input data or previous layer.
Ultimately, we gain the useful property of equivariance for image analysis by using filters. This means that a change in the input data will lead to a similar change in the output data. This makes the network more resilient to changes in the input, e.g. if the images are taken from a different angle.

Detector stage: The activation maps are run through a preferably non-linear activation function in this stage, introducing non-linearity in the model. The typical function employed here is the sigmoid function as given in equation (3.2).

Pooling stage: In this layer the input is downsampled, by combining adjacent rectangular regions into singular values. One can use either the max value, the average value or a different value calculated from the the given input, such as the L2 norm. Pooling is used to identify properties in the data, that do not rely on their exact position, but rather on their general location or existence at all [12].

Last layer: Usually the last layer is a fully connected layer, not only for binary cases, but also for multi-class classifications.
As well as in FFNNs a loss function needs to be defined, which is minimized in an iterative process by updating the weights and biases in each layers through the backpropagation algorithm and recalculating the loss. For binary cases the cross-entropy function is used here as well, which is given in equation (3.3).

3.3.1 | The U-net architecture for image segmentation

Segmenting medical images into morphological regions is an area of active research, commonly used for extracting quantitative imaging parameters e.g. using the Radiomics paradigm (section 3.1) or recognized clinical parameters (e.g. mean Houndsfield units from x-ray / computed tomography or max standardized uptake values from positron emission tomography images). A fully connected CNN-architecture proposed

in 2016 having shown clinical promise for segmenting medical images is known as U-net, using a symmetric shape of convolutional layers encoding the images to low-resolution layers before decoding back up to the original resolution for predicting the probability of each voxel belonging in a volume of interest (VOI) [13]. While U-net have shown potential to segment both tumour lesions and a large number of anatomical regions (e.g. organs, or subdivided organ parts) clinical implementation is still a work in process to evaluate the segmentation performace relative clinically trained radiologist, but ensures a lower inter-observer variability and thus a larger reproducibility [14].

3.4 | Logistic Regression

Logistic regression is one of the simplest models used for classification tasks with discrete inputs, most commonly applied onto binary classification problems. In this case usually the sigmoid function (3.2) is used to fit the given dataset, which is defined between 0 and 1 and is used for estimating the probability of the possible outcome. This is done by calculating the parameters giving the highest probability to explain the training data, which is also known as the maximum likelihood estimation [15]. To estimate the parameters and train the model a cost function is minimized, which in case of a binary case is usually the cross-entropy function (3.3). This is typically done using a gradient descent method, iteratively improving the model parameters. It is often also beneficial to include either L1, or L2-regularization terms in the cost function. Since the sigmoid function only returns a probability for the possible outcome, a decision boundary needs to be chosen, which picks a specific binary case, if the value is above or below the boundary.

3.5 | Random trees

Decision trees can be of good use e.g. for identifying important features in classification tasks. For machine learning purposes Breiman et al. first introduced the idea of CART (Classification and regression tree) [16]. Using the metaphor of trees, each conjunction in a tree divides the dataset into two branches based on a given feature. After a given amount of conjunction the dataset is split up into various branches, which in the end turn into leaves representing the class labels. While the first conjunctions focus on diving the dataset according to the most decisive features, later conjunctions are less of importance. Relying on excessive amount of conjunctions will lead to overfitting. It is thus difficult to decide when to stop splitting into different branches.

3.6 | Random forests

The idea of random forests was first introduced by Tin Kam Ho as a way to deal with the issues within decision trees being difficult to generalize for the testing set and overfitting is easily achieved [17]. Random forests as used nowadays create a set of different decision trees and sample their results. To ensure a variety of trees, Bootstrap aggregation or in short Bagging is used, training each tree with a different subset of data from the training set. At each split in a decision tree a random subset of features is selected for taking the split decision, which also improves the diversity among the trees. For classification tasks a simple vote among all trees is considered for the decision, while for regression tasks the average is usually taken. The diversity among trees reduces the risk of overfitting.

3.7 | Gradient Boosting

Boosting methods in general try to combine multiple ”weak learners” such as decision trees into one ”strong learner”. In an iterative process each model of a weak learner tries to improve based on the error of the previous one. Therefore, a loss function is required, which could be the mean squared error in regression tasks or the cross-entropy function in classification tasks. Similar to the gradient descent method the derivative of the loss function is calculated with respect to prediction given by the previous model and is used as a weight for the next ”weak learner” to improve the previous model. This is repeated for a given amount of iterations. Instead of taking an entire decision tree, usually a tree with a very limited amount of branches is used. This way, Gradient boosting often achieves better results than decision trees or random forest, but in turn looses the easiness to interpret the result and isolate most important features.
One library employing gradient boosting is the extreme gradient boosting, or XGBoost library, which also uses decision trees as weak learners. It is known for efficient computation and good accuracy.

3.8 | Evaluation

The different models are evaluated based on a set of different scores.
Accuracy: Accuracy describes the percentage of correctly classified images.

Brier Score: The Brier score is quite similar to the mean squared error, but applied onto classification problems, or in this case binary classification. It calculates the square of the difference of the estimated probability of one value y_{pred} (between 0 and 1) to the actual result y (0 or 1). For a series of values it takes the average over all squares. Thus it is defined as

$$BS = \frac{1}{N}\sum_{i=1}^{N}(y_{pred}^{i} - y) \tag{3.4}$$

Recall: The Recall gives the percentage of correctly predicted positives from all actual positive cases, it is defined as

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}. \tag{3.5}$$

Precision: Similarly to the recall, the precision defines the percentage out of all positive classified cases, how many were correctly classified, thus it is given by

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}. \tag{3.6}$$

F1: The F1 score is the harmonic mean of the recall and the precision and is thus defined as

$$\text{F1} = \frac{2 \cdot \text{True Positive}}{2 \cdot \text{True Positive} + \text{False Positive} + \text{False Negative}}. \tag{3.7}$$

4 | Materials and methods

For this project we are using a set of 21 333 chest radiographs for possible pneumonia detection, which were presented as a Kaggle challenge in 2018 with images provided by Radiological Society of North America (RSNA®), the US National Institutes of Health, the Society of Thoracic Radiology, and MD.ai [18]. Of the images 16 569 (77.7 %) were of healthy subjects, while 4764 (22.3 %) were of patients diagnosed with pneumonia.

4.1 | Pre-processing

Prior to training, the images underwent prepprocessing using PyTorch's torchvision package and the pydicom library. The images were first converted to PNG-grayscale images. The images were then preprocessed by uniformly resizing them to 256x256 pixels for standardization across the dataset- The image pixel values were then normalized to the range -1 to 1.

4.2 | Hand-crafted radiomic features

4.2.1 | ROI segmentation and extraction

Hand-crafted radiomic features were extracted from the x-ray radiographs using the pyRadiomics python package [7]. 1023 radiomic features were extracted per image.
To segment the lungs single region of interest (ROI) for feature extraction, a pre-trained U-net CNN (see section 3.3.1) found on GitHub was applied to the radiographic images [19]. A pixel was included in the ROI if the U-net CNN predicted the probability of the pixel being a part of the lung tissue with certainty above 0.9 (i.e. using thresholding on the predicted class probability).

Shape-features were calculated using only the ROI's. First-order and textural features were extracted from the ROI's without any pre-processing filter, in addition to after applying 7 different filters before extraction (logarithm of gradient, wavelet, square, squareroot, logarithm, exponential, gradient). Before extracting textural features the images were discretized into 32 gray-levels (GL) using the fixed bin count (FBC) approach, only including gray-levels from the ROI's. Textural features were calculated from 5 texture matrices: the co-occurance matrix (GLCM), run-length matrix (GLRLM), zone-size matrix (GLZSM), dependency matrix (GLDM), neighbourhood gray-tone difference matrix (NGTDM).

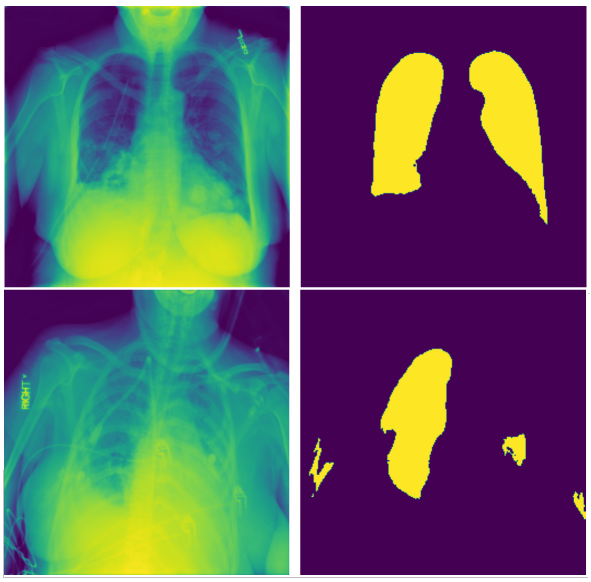

Figure 4.1: Lung regions predicted by the U-net inspired CNN, applied on the downsampled radiographic images (left). While the CNN performed well by visual inspection on most of the images (top row), it was less successful on others - including some obvious non-lung regions while omitting parts of the lungs (bottom row).

4.2.2 | Feature selection and modelling

Four different model types were used to predict which images were of patients having pneumonia: logistic regression (LR), random forest (RF), XGBoost (XGB), and a dense feed-forward neural network (FFNN). The radiomic features were normalized using standard-scaling, using the mean and standard deviation for each feature - found using the training set only to avoid data leakage [9].
The LR was penalized with a l1-emphasized regularization (0.9 l1-ratio) with a penalization strength of 1.0. The RF model was setup to minimum require 10 sample to create a new split, with a maximum depth of 20 features. No hyperparameter tuning was performed for the XGBoost.
To reduce the number of radiomic features used in the final models a recursive feature elimination (RFE) was performed for each of the three non-deep models, removing the 10 least important features across a 3-fold validation for each iteration. The optimal number of features was determined on a per-model basis by maximizing the cross-validated f1-score.

4.2.3 | Combining radiomic features with a feed-forward neural network (FFNN)

In a regular FFNN a layer structure with two hidden layers containing 8 nodes each was used for creating a model using the radiomic features as input. The Relu function was used as activation function in the hidden layers, while the sigmoid function was used as the output function. Both hidden layers were constricted using L2-regularization. In a first try using only 10 epochs it was attempted using RMSprop as well as Adam as gradient methods, while for a longer attempt with 1000 epochs only Adam was used. The program was written using the Python package Keras.

4.3 | CNN

4.3.1 | Model Architecture

We used transfer learning, starting with the VGG16 architecture. The VGG16 architecture utilized in this study comprises a series of convolutional layers followed by max-pooling operations and fully connected layers. The model starts with a stack of two convolutional layers with a small 3x3 kernel size, each employing 64 filters and utilizing the ReLU activation function. Subsequently, max-pooling layers with a 2x2 window and stride 2 are applied to reduce spatial dimensions. This pattern repeats multiple times, forming the convolutional feature extraction section. The deeper layers consist of three sets of double convolutional layers, where each set contains two successive 3x3 convolutions with 128, 256, and 512 filters, respectively, followed by max-pooling. The model concludes with three fully connected layers, the first two comprising 4096 units each with ReLU activation and a dropout probability of 0.5 for regularization. The final layer is a linear transformation mapping the 4096-dimensional vector to the output of 2 classes, employing a softmax activation function for classification.

4.3.2 | Training and evaluation

For training, to prevent data leakage, the training set was further sub-divided into a training and validation-set with a 80-20 split, using stratification to ensure that classes were equally distributed.

4.3.3 | Fine-Tuning Strategy

To adapt the pretrained VGG16 model for the specific binary classification task, a transfer learning approach was implemented. Firstly, the parameters of the convolutional layers, i.e., the `features` part of the VGG16 architecture, were frozen to prevent updates during the initial stages of training. This was achieved by iterating through the `model_pre.features.parameters()` and setting `required_grad` to `False` for each parameter.
Following this, adjustments were made to the classifier section of the VGG16 model. The number of input features for the last fully connected layer was determined using `model_pre.classifier[6].in_features`. Subsequently, the final fully connected layer was replaced with a new `nn.Linear` layer that maps to the number of classes in the dataset (`len(class_names)`). The classifier was modified using a sequential arrangement of layers formed by appending the new linear layer to the existing layers, effectively adjusting the output size for the specific classification task. This adaptation process was implemented through the code snippet:

```
num_features = model_pre.classifier[6].in_features
features = list(model_pre.classifier.children())[:-1]
features.extend([nn.Linear(num_features, len(class_names))])
model_pre.classifier = nn.Sequential(*features)
```

After configuring the VGG16 model for transfer learning, the model was transferred to the designated computational device, either a CPU or GPU. To optimize the model's performance on the binary classification task, the criterion for loss computation was set to cross entropy loss.
For training the model, an SGD optimizer with a learning rate of 0.001, momentum of 0.9, and weight decay of 0.01 was employed. To enhance training efficiency, a learning rate scheduler was utilized. This scheduler adjusted the learning rate by a factor of 0.1 every 10 epochs.
The non-segmented radiomic features (all images used but without segmenting a ROI) was evaluated both on the same train / test split as for evaluating the CNN, but also on a secondary split configuration for comparing the U-net segmented radiomic features.

5 | Results

5.1 | Radiomics

Model	N_{fts} used	Acc %	Precision %	Recall %	F1 %	Brier
Logistic Regression	93	82.4	65.0	42.4	51.3	0.18
Random Forest	103	81.8	64.5	37.4	47.4	0.18
XGBoost	503	81.0	66.2	39.8	47.8	0.19
Feed-forward neural network	1023	81.1	63.2	30.3	41.0	0.13

Table 5.1: Final scores for all models using the hand-crafted radiomic features. The optimal number of features were found for each model using RFE.

The optimal number of radiomic features used in the final models, found using RFE with 3-fold CV, ranged between 93 for the LR and 503 for the XGBoost as seen in table 5.1 and figure 5.1. No feature selection was performed for the FFNN. Results for the LR, RF, and XGB models without applying feature selection before training is found in table A.1 in the Appendix.
While LR had the highest accuracy at 82 %, the precision was slightly higher for XGB. The FFNN had a significantly lower brier score than the non-deep models, but had the lowest precision and recall among all the radiomic-based models.

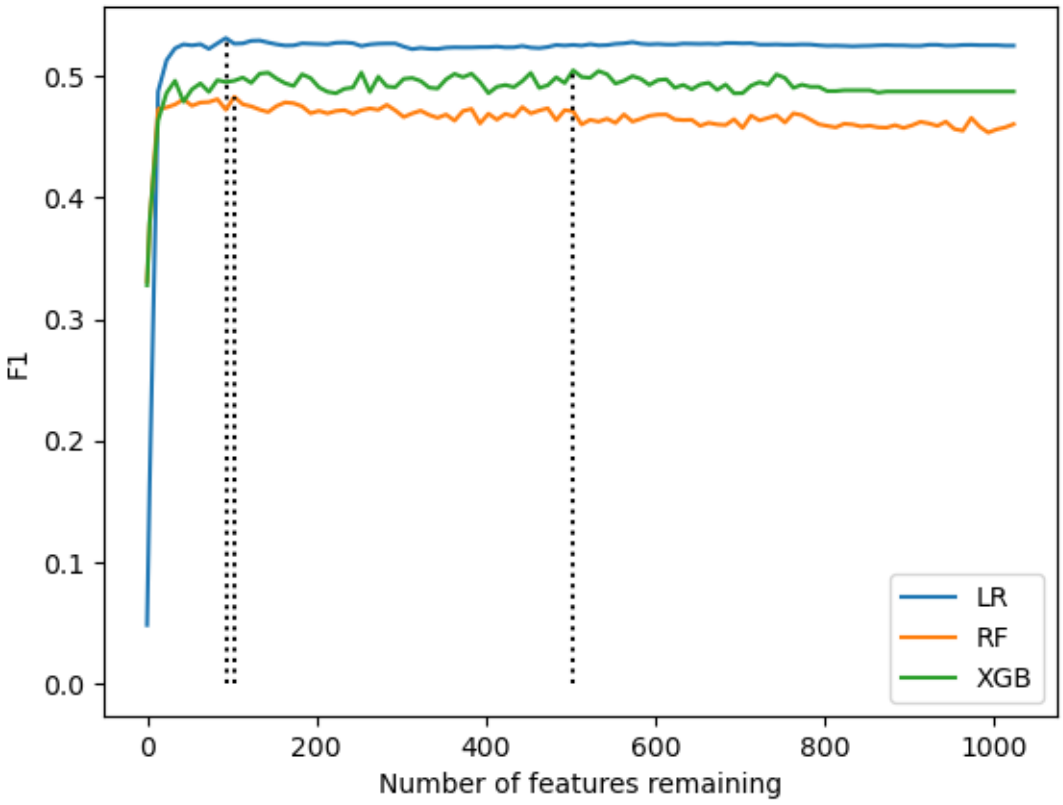

Figure 5.1: Selecting the optimal number of features for each model using hand-crafted radiomic features, by maximizing the averaged f1-score across a 3-fold CV.

5.1.1 | Using FFNN on radiomic features

For first try using RMSprop and only 10 epochs this gradient method lead to predicting only True cases, thus not differentiating in between the images. Thus the gradient method was switched to Adam, which yielded similar results in accuracy. This is shown for the grid search employed in both cases in figure A.1 in the Appendix. However, since the grid search was only performed on fewer 279 features, instead of the

entire set of 1023 features and less images for reducing the computational time, the actual results for the RMSprop were worse.
The Adam program however yielded for a regularization parameter of 0.01 and a learning rate of 0.1 an overall accuracy of 81.09%. The other evaluation parameters are shown in table 5.1.

5.2 | CNN

Due to time constraint the transfer-learning CNN was trained for five epochs. Figures 5.2 and 5.3 show the accuracy and loss for training and validation, where the validation is 20 percent of the training data (not the final test set). On the final test set an accuracy score of 76% was achieved, with a recall of 13.8% calculated from the resulting confusion matrix on the test set shown in figure 5.4.

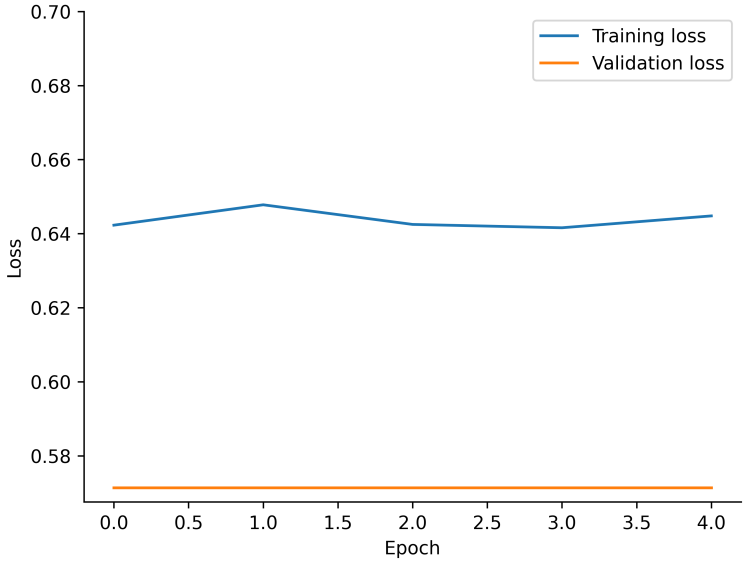

Figure 5.2: Training and validation loss over epochs. The plot illustrates the progression of loss values for both training and validation datasets across epochs during the model training process.

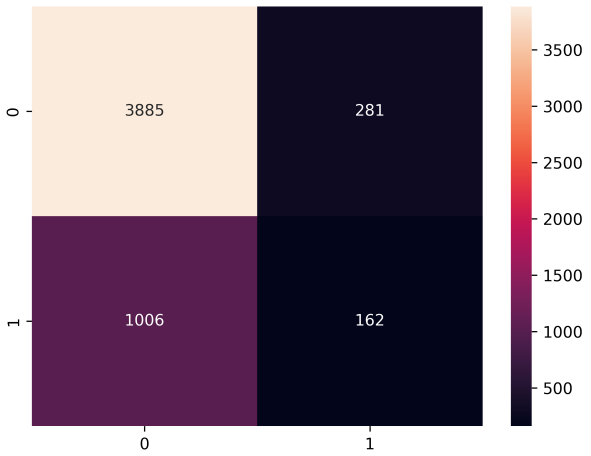

Figure 5.4: Heatmap representation of the confusion matrix of the CNN. Each cell visualizes the frequency or intensity of predictions made by the model compared to the ground truth across different classes. Darker shades indicate higher values, highlighting areas of accurate predictions (diagonal) and misclassifications (off-diagonal).

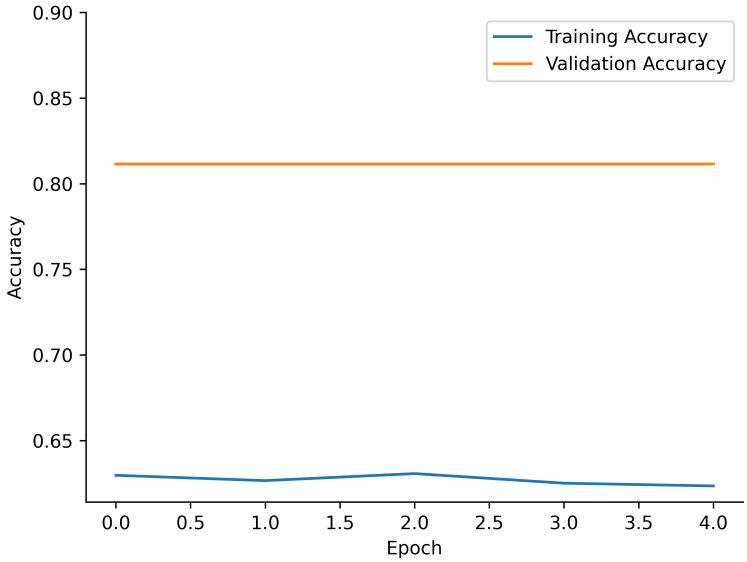

Figure 5.3: Training and validation accuracy over epochs. This plot depicts the evolution of accuracy scores for training and validation datasets throughout the epochs during the model training phase.

6 | Discussion

This classification problem is highly studied with numerous solutions having been published during the last decade [20] using CNNs alone. In the current work we used a transfer learning implementation of a CNN to compare with the radiomics-driven classification. Although modest results were acquired, the model should be ultimately compared to recent findings in the literature, of which a relatively high accuracy and recall (above 0.9) is reached. According to a recent review of CNNs to characterize X-ray images, one of the most powerfull solutions was the Inception V3-model, a CNN with 42 layers [20].
All the radiomic-based models, as well as the CNN, is seen to have a high amount of false negatives (overall low recall). One might expect the somewhat high ratio of healthy subjects to pneumonia patients in the data (about 3.5 healthy per pneumonia patient) have had an impact on what the models deemed important during training, at least when optimizing for accuracy.
However, the combination of radiomic features with Random Forest, XGBoost, and FFNN has shown some ability to distinguish pneumonia from healthy cases in X-ray images. This aligns with the foundational principles of radiomics, which share similarities with texture analysis in satellite data. The correlation between the representation of lungs in X-ray images and texture analysis in satellite data supports the effectiveness of applying established techniques from related domains, enhancing the reliability of our approach in medical imaging analysis.

6.1 | Model responses to feature selection

While applying feature selection improved upon most model scores for the models, the improvement was marginal. While the f1-score might seem to slightly increase with less features for the RF, no pattern is obvious among the model responses to feature selection in figure 5.1 - which seem to be more or less randomly fluctuating. Due to the random nature of both RF and XGB, the increased variability in f1-score relative LR seem reasonable - and might reflect that the models either are able to handle feature selection internally (using LR l1-penalization, RF tree depth, etc) such that applying RFE is redundant.

6.2 | Improvements for further studies

There are numerous way to improve the CNN, simply running for more epochs might lead to a more favorable results. Also, there might be an introduction of abnormal behaviour stemming from class imbalance that has to be resolved. An implementation with weights was attempted but not finished due to time constraints.

The extraction of hand-crafted radiomic features leaves room for many choices in the process, such as the discretization choice and number of bins / bin width or parameters for each of the pre-extraction filters. The segmentation U-net could have been improved upon, and the predicted ROI's validated by a trained radiologist in a blinded study.
The implementation of the FFNN could be improved in various ways. First of different activation functions could be tested, as well as a grid search based on all radiomic parameters could be employed. Since one layer structure was tested also with only very few nodes one could also test larger networks and whether they could improve accuracy. However, the focus of this project was not directed on the FFNN, it is mostly used to compare to the CNN.

7 | Conclusions

Different ways to classify X-ray images were explored and compared: both hand crafted radiomic features, and a deep convolutional neural network. Models using the radiomic features resulted in accuracy of slightly above 80 percent. The tested methods involved logistic regression (82.4%), random forests (81.8%), XGBoost (81.0%) and FFNN (81.1%).
The CNN structure achieved an accuracy of 76%. Since other works achieve accuracies over 90% with this method the CNN could have been further improved in numerous ways, mainly using more epochs to train the network. Overall the data imbalance seem to have lead all models to overly represent the healthy subjects while neglecting pneumonia cases, as the recall for all models never went above 42.4 % while reaching as low as 13.8% for the CNN.

References

[1] Nasreen Mahomed et al. "Preliminary report from the World Health Organisation Chest Radiography in Epidemiological Studies project". In: *Pediatric Radiology* 47.11 (2017), pp. 1399–1404. DOI: 10.1007/s00247-017-3834-9.

[2] Muhammad Mujahid et al. "Pneumonia Classification from X-ray Images with Inception-V3 and Convolutional Neural Network". In: *Diagnostics* 12 (2022), p. 1280. DOI: 10.3390/diagnostics12051280.

[3] Rogers et al. "Radiomics: from qualitative to quantitative imaging". In: *The British Journal of Radiology* (2020). DOI: 10.1259/bjr.20190948.

[4] Zwanenburg et al. "The Image Biomarker Standardization Initiative: Standardized Quantitative Radiomics for High-Throughput Image-based Phenotyping". In: *Radiology* (2020).

[5] Tixier et al. "Intratumor heterogeneity characterized by textural features on baseline 18F-FDG PET images predicts response to concomitant radiochemotherapy in esophageal cancer". In: *Journal of Nuclear Medicine* (2011).

[6] Leijenaar et al. "The effect of SUV discretization in quantitative FDG-PET Radiomics: the need for standardized methodology in tumor texture analysis". In: *Scientific Reports* (2015).

[7] van Griethuysen et al. "Computational Radiomics System to Decode the Radiographic Phenotype". In: *Cancer Research* (2017). DOI: 10.1158/0008-5472.CAN-17-0339.

[8] Mehta et al. *A high-bias, low-variance introduction to Machine Learning for physicists*. Physics Reports, 2019, p. 63.

[9] Maleki et al. "Generalizability of Machine Learning Models: Quantitative Evaluation of Three Methodological Pitfalls". In: *Radiology: Artificial Intelligence* (2023). DOI: 10.1148/ryai.220028.

[10] Guyon et al. "Gene Selection for Cancer Classification using Support Vector Machines". In: *Machine Learning* (2002). DOI: 10.1023/A:1012487302797.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[12] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[13] Shelhamer et al. "Fully Convolutional Networks for Semantic Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2016). DOI: 10.1109/TPAMI.2016.2572683.

[14] Siddique et al. "U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications". In: *IEEE Access* (2021).

[15] Kevin P. Murphy. *Probabilistic Machine Learning: An Introduction*. MIT Press, 2022, p. 109. URL: https://probml.ai.

[16] Leo Breiman et al. "Classification and Regression Trees". In: *Wadsworth International Group* (1984).

[17] Tin Kam Ho. "Random decision forests". In: *Proceedings of the 3rd International Conference on Document Analysis and Recognition* 1 (1995), pp. 278–282.

[18] MD Anouk Stein et al. *RSNA Pneumonia Detection Challenge*. 2018. URL: https://kaggle.com/competitions/rsna-pneumonia-detection-challenge.

[19] Vitali Petsiuk. *Lung fields segmentation on CXR images using convolutional neural networks*. 2020. URL: https://github.com/imlab-uiip/lung-segmentation-2d.

[20] DJ Alapat, MV Menon, and S Ashok. "A Review on Detection of Pneumonia in Chest X-ray Images Using Neural Networks". In: *Journal of Biomedical Physics & Engineering* 12.6 (Dec. 2022), pp. 551–558. DOI: 10.31661/jbpe.v0i0.2202-1461.

A | Appendix

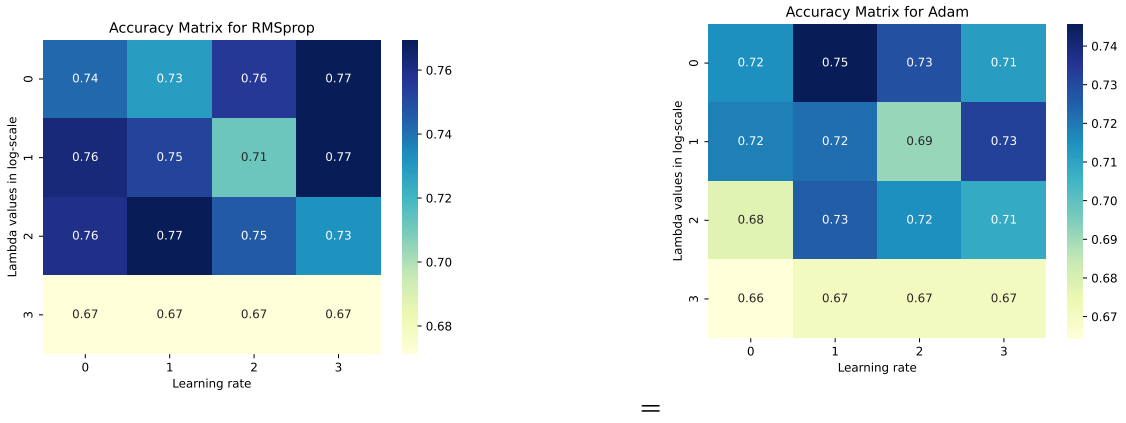

Figure A.1: The grid search was first employed on a smaller dataset suing less radiomics features and fewer epochs to determine adequate parameters. In this case the RMSprop seemed to perform better, however if used on all features the accuracy dropped, not differentiating the cases anymore. Both learning rate as well as the L2-regularization parameter were used in log scale from 10^{-3} up to 1, with the number in the graph depicting the negative exponential number.

Model	Acc %	Precision %	Recall %	F1 %
Logistic Regression	82.8	64.8	40.6	49.9
Random Forest	81.7	65.0	35.2	45.7
XGBoost	81.1	66.2	39.7	47.9

Table A.1: Final scores for all models using the hand-crafted radiomic features, without any feature selection applied before training and testing the models.