



UNIVERSITEIT•STELLENBOSCH•UNIVERSITY  
jou kennisvennoot • your knowledge partner

# **Tuberculosis Detection with Cough Audio and Metadata Analysis**

Paul Roux Ellis  
23790288

Report submitted in partial fulfilment of the requirements of the module  
Project (E) 448 for the degree Baccalaureus in Engineering in the Department of  
Electrical and Electronic Engineering at Stellenbosch University.

Supervisor: Prof T. R. Niesler

October 2023

# Acknowledgements

I would like to express my gratitude to the following individuals for their support and guidance throughout this project:

- My supervisor, Prof T.R. Niesler, for all your guidance and support throughout the project, your feedback was instrumental.
- My parents and brother, Gideon, Paulette and Philip, without whom I would not have this opportunity. Thank you for all the support throughout this project and my entire degree.
- The members of the DSP lab and my fellow skripsie students, Minette and Rachel. The members of the DSP lab for being able to help with difficult questions, and Minette and Rachel for constant motivation and encouragement throughout this project.
- Lastly to all my friends, who spent many hours providing support during this project.



UNIVERSITEIT • STELLENBOSCH • UNIVERSITY  
jou kennisvennoot • your knowledge partner

## Plagiaatverklaring / *Plagiarism Declaration*

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.

*Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.*

2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.

*I agree that plagiarism is a punishable offence because it constitutes theft.*

3. Ek verstaan ook dat direkte vertalings plagiaat is.


*I also understand that direct translations are plagiarism.*

4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelike aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.

*Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism*

5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.

*I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.*

23790288 Studentenommer / <i>Student number</i>	 Handtekening / <i>Signature</i>
PR Ellis Voorletters en van / <i>Initials and surname</i>	06/11/2023 Datum / <i>Date</i>

# Abstract

## English

Tuberculosis (TB), the second leading infectious killer globally in 2021, necessitates an early diagnosis for effective treatment and to reduce the mortality rates. The availability of rapid molecular diagnostic tests remains limited in developing countries. This report explores the feasibility of using cough audio and clinical data obtained from a patient survey to classify patients as TB-positive or TB-negative. Signal processing techniques are used to extract relevant features from the cough audio, and used as input to baseline Logistic Regression (LR) models and different Convolutional Neural Network (CNN) architectures. Hyperparameter optimisation is performed on the feature extraction, as well as both the LR and CNN architectures to find the best-performing architecture. The findings indicate that on a per-cough audio-only assessment, the CNN achieved an AUC of 0.771 and on a per-participant assessment an AUC of larger than 0.830 was achieved by all the different architectures with metadata included. The findings indicate the potential for using short cough audio recordings for tuberculosis diagnosis.

## Afrikaans

Tuberkulose (TB), die aansteeklike siekte wat die tweede meeste steftes wêreldwyd in 2021 veroorsaak het, vereis 'n vroeë diagnose om effektiewe behandeling te kry en sterftesyfers te verminder. Die beskikbaarheid van vinnige molekulêre diagnostiese toetse in ontwikkelende lande is beperk. Hierdie verslag ondersoek die moontlikheid om pasiënte as TB-positief of TB-negatief te klassifiseer deur hoesopnames en kliniese data van pasiëntopnames te gebruik. Seinverwerkingstegnieke word gebruik om relevante kenmerke uit die hoesopnames te onttrek en as insette vir Logistiese Regressiemodelle (LR) en verskeie Konvolusionele Neurale Netwerk (KNN) argitekture te gebruik. Optimering word uitgevoer op die kenmerk-onttrekking, asook op beide die LR- en KNN- argitekture om die beste argitektuur te vind. Die resultate dui daarop dat op 'n hoesopname, die KNN 'n oppervlakte onder die kurwe van 0.771 bereik, en op 'n per pasiënt beoordeling 'n oppervlakte onder die kurwe van meer as 0.830 deur al die argitekture behaal word. Die resultate dui aan op die potentiaal om kort hoesopnames te gebruik vir tuberkulose-diagnose.

# Contents

<b>Declaration</b>	<b>ii</b>
<b>Abstract</b>	<b>iii</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>ix</b>
<b>Nomenclature</b>	<b>xi</b>
<b>1. Introduction</b>	<b>1</b>
1.1. Motivation . . . . .	1
1.2. Scope . . . . .	1
1.3. Structure . . . . .	2
<b>2. Literature Review</b>	<b>3</b>
2.1. Detection of tuberculosis by automatic cough sound analysis [1] . . . . .	3
2.2. Automatic Cough Classification for Tuberculosis Screening in a Real-World Environment [2] . . . . .	3
2.3. Predicting Tuberculosis from Real-World Cough Audio Recordings and Metadata [3] . . . . .	4
2.4. COVID-19 detection in cough, breath, and speech using deep transfer learning and bottleneck features [4] . . . . .	4
2.5. Development and clinical validation of Swaasa AI platform for screening and prioritization of pulmonary TB [5] . . . . .	5
2.6. Deep Neural Network-Based Respiratory Pathology Classification Using Cough Sounds [6] . . . . .	5
2.7. Deep recurrent neural networks with attention mechanisms for respiratory anomaly classification [7] . . . . .	6
2.8. Conclusion . . . . .	6
<b>3. Background</b>	<b>7</b>
3.1. Features . . . . .	7
3.1.1. Log Spectral Energies . . . . .	7
3.1.2. Mel-Frequency Cepstral Coefficients (MFCCs) . . . . .	7

3.1.3. Zero-crossing rate . . . . .	9
3.1.4. Kurtosis . . . . .	9
3.2. Models . . . . .	9
3.2.1. Logistic Regression . . . . .	10
3.2.2. Convolutional neural network (CNN) . . . . .	10
3.3. Conclusion . . . . .	13
<b>4. Data</b>	<b>14</b>
4.1. Data collection procedure . . . . .	14
4.2. Ground truth TB testing . . . . .	15
4.3. Dataset summary . . . . .	17
4.4. Training, development and test set . . . . .	17
4.5. Conclusion . . . . .	17
<b>5. Experimental setup</b>	<b>20</b>
5.1. Evaluation Metrics . . . . .	20
5.1.1. Receiver Operating Characteristic (ROC) . . . . .	20
5.1.2. Area under the curve . . . . .	22
5.1.3. Youden Index . . . . .	22
5.1.4. Per-cough and per-participant assessment . . . . .	22
5.1.5. Software tools . . . . .	22
5.2. Feature Extraction . . . . .	23
5.2.1. Framing . . . . .	23
5.2.2. MFCCs and mel spectrogram . . . . .	23
5.2.3. Metadata . . . . .	24
5.2.4. Feature extraction hyperparameters . . . . .	25
5.3. Hyperparameter tuning . . . . .	25
5.3.1. Logistic Regression . . . . .	25
5.3.2. Convolutional Neural Network . . . . .	27
5.4. Conclusion . . . . .	32
<b>6. Experimental Results</b>	<b>33</b>
6.1. Logistic Regression . . . . .	33
6.2. Convolutional Neural Network . . . . .	33
6.3. Summary . . . . .	34
6.4. Conclusion . . . . .	36
<b>7. Discussion and Conclusion</b>	<b>37</b>
7.1. Main findings . . . . .	37
7.2. Future Work . . . . .	38

<b>Bibliography</b>	<b>39</b>
<b>A. Project Planning Schedule</b>	<b>42</b>
<b>B. Outcomes Compliance</b>	<b>43</b>

# List of Figures

3.1.	Example of a logarithmically distributed triangular filterbank. . . . .	8
3.2.	Visual illustration of applying a filter/kernel to an input image. This process is applied to each possible filter overlap on the input image resulting in a 2x2 output image. In this example $N_{x1} = N_{x2} = 3$ and $N_{h1} = N_{h2} = 2$ . . .	11
3.3.	The basic structure of a CNN which includes the 3-layered input image which contains the MFCCs, its first and its second differentials. It shows one convolutional layer with a pooling layer which can then be extended to multiple layers. The last layer is the fully connected layer which maps the output to two different classes for which a softmax activation function is applied to obtain the probabilities of the input belonging to a certain class. Adapted from [8]. . . . .	12
5.1.	Confusion matrix illustrating the classification results with true positives (TP) and true negatives in green, false positives (FP) and false negative (FN) in red. The matrix depicts the relationship between actual positive and negative instances versus predicted positive and negative outcomes, providing insights into the performance of a classification model. . . . .	21
5.2.	An example of a receiver operating characteristic (ROC) curve. . . . .	21
5.3.	A mel spectrogram with 128 filters and MFCCs with 39 coefficients extracted from a positive TB cough. . . . .	24
5.4.	The development set ROC curve of the top-performing audio-only logistic regression model using 13 MFCCs extracted from 1024-point frames that overlap by 25% is shown on the left. On the right, the development and training losses are plotted as the model was trained. The point where the development loss levels off indicates the start of overfitting, which is why early stopping was implemented. . . . .	27
5.5.	ROC curves of the participant which compares the cough-only experiment with the cough and metadata experiment. . . . .	28
5.6.	AUC scores for convolutional neural networks trained with two, three, and four convolutional layers showed varying AUC distributions. Larger CNNs had a higher variance due to the larger number of trainable parameters. . .	30



5.7.	Architecture of the best performing CNN model with the optimal hyperparameters tuned on the development set. It shows the different convolutional and fully connected layers with the dropout rate used. . . . .	31
5.8.	AUC scores for convolutional neural networks trained with two, three, and four convolutional layers showed varying AUC distributions. Larger CNNs had a higher variance due to the larger number of trainable parameters. . .	31
6.1.	Test set ROC curves for per-cough and per-participant classification. For per-cough classification, the best performing model was the two convolutional-layered CNN using the three-layered MFCCs as input. For per-participant classification, the best model was the metadata combined with the logistic regression results. . . . .	36

# List of Tables

4.1.	Audio recording lengths from participants. . . . .	15
4.2.	Table of TB participant data. The columns "TB+" and "TB-" list the number of patients who were TB positive and negative respectively. The quantities $\mu$ and $\sigma$ are the mean and standard deviation of the cough audio samples. The final two columns indicate the number of coughs recorded. . . . .	17
4.3.	The metadata gathered for each participant in Table 4.2. . . . .	18
4.4.	Table of the data distribution between the train, development and test splits. The first column of each set is the number of TB-positive patients and coughs per set. The second column of each set describes the number of TB-negative patients and coughs per set. The final % column is the percentage of positive patients or coughs per set. . . . .	19
5.1.	Table of the number of frames from the audio for each different % overlap between successive frames. . . . .	23
5.2.	Feature extraction hyperparameters and the ranges considered. . . . .	25
5.3.	Different hyperparameters for logistic regression and the range on what optimisation techniques were performed. . . . .	26
5.4.	Logistic regression hyperparameter tuning results. The best result for each combination of % overlap, number of MFCCs and $\lambda$ are shown in the table based on the performance on the development set. . . . .	26
5.5.	Table of the different constant and tuned hyperparameters used in the experimental process for CNNs. A short description as well as the range of the different hyperparameters are listed. . . . .	29
5.6.	Table of the best hyperparameters found during the grid search for each of the different models as measured on the development set. . . . .	32
6.1.	Performance of the logistic regression models with optimised hyperparameters on the development and on the test set. Results are shown for (a) audio-only cough classification (b) audio-only participant classification and (c) audio and metadata per-participant classification. . . . .	34

6.2. Performance of the CNN models with optimised hyperparameters on the development and on the test set. Results are shown for (a) audio-only cough classification (b) audio-only participant classification and (c) audio and metadata per-participant classification . . . . .	35
6.3. Best classifier for the audio-only per-cough classification, audio-only per-participant classification and audio and metadata per-participant classification.	36
B.1. ECSA outcomes compliance . . . . .	43

# Nomenclature

## Variables and functions

$\mu$	Sample mean
$\sigma$	Standard deviation
$N$	Number of samples
$\sigma(z)$	Sigmoid function
$\hat{y}$	Probability of input belonging to the positive class
$\alpha$	Vector of feature weights
$\lambda$	L2 Regularisation strength parameter
$r_{xh}$	Output image from kernel swept over input image in CNN
$h$	Kernel of convolutional layer
$L(y, \hat{y})$	Binary cross entropy loss
$\eta$	Learning rate
$\beta_1$	Exponential decay rate for first moment in Adam
$\beta_2$	Exponential decay rate for second moment in Adam
$\delta$	The % overlap between successive frames of cough audio
$\mathcal{M}$	The number of MFCCs
$\mathbb{D}$	Dropout rate of dropout layer
$B$	Batch size
$l$	Number of convolutional layers

**Acronyms and abbreviations**

A-BiLSTM	Attention-based Bidirectional Long Short-Term Memory Network
ANN	Artificial Neural Network
AUC	Area under curve
CNN	Convolutional Neural Network
COPD	Chronic Obstructive Pulmonary Disease
COVID-19	Coronavirus Disease of 2019
DCT	Discrete Cosine Transform
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
LR	Logistic Regression
LSTM	Long Short-Term Memory
MGIT	Mycobacteria Growth Indicator Tube
MFCCs	Mel-Frequency Cepstral Coefficients
MLP	Multilayer Perceptron
MTB	Mycobacterium tuberculosis
R2D2	Rapid Research and Diagnostics Development
RIF	Rifampin
ROC	Receiver Operating Characteristic
TB	Tuberculosis
TPR	True Positive Rate
TN	True Negative
TP	True Positive
WHO	World Health Organisation
ZCR	Zero-crossing rate

# Chapter 1

## Introduction

### 1.1. Motivation

Tuberculosis is the second leading infectious killer after COVID-19 worldwide in 2021 [9]. In 2021, a worldwide estimated total of 10.6 million people fell ill with tuberculosis, of these 1.6 million people died [9]. Early diagnosis of tuberculosis is essential for correct treatment which will reduce deaths and improve the rates of recovery. The current techniques recommended by the World Health Organisation (WHO) for diagnosing tuberculosis include rapid molecular diagnostic tests as an initial test for people experiencing any symptoms of tuberculosis [9]. Xpert MTB/RIF Ultra and Truenat assays form part of the rapid tests recommended by the WHO for their high diagnostic accuracy [9]. However, especially in developing countries, the funding and the correct equipment to perform these tests are often not available. There is therefore a clear need for a low-cost screening test in countries which does not require any specialized equipment or laboratory facilities.

Studies have shown that acoustic analysis of cough sounds, which are one of the main symptoms of tuberculosis, can be used to help classify the disease [1], [2] [3] and [5].

### 1.2. Scope

In this work, we aim to build classifiers for two different datasets collected together and from the same patients. Firstly, a set of cough-audio recordings was compiled. Secondly, metadata that includes other measurements and indications of the patient's health was collected. These complementary datasets can be used both in isolation and together to make a classification decision regarding whether the patient has TB. Signal processing techniques are used to extract different features from the cough audio which will be used in the models. Experiments will be done to compare the differences in both classifiers to find the best model for predicting tuberculosis from cough audio. Several convolutional neural network architectures will undergo testing and be evaluated on the hold-out test set. The chosen best architecture will then be explained and discussed in the context of the field.

## 1.3. Structure

The report is structured as follows:

- Chapter 2 provides a review of the literature in the field of cough-audio classification. The different findings in each study are discussed, and relevant aspects are highlighted which are used in this study.
- Chapter 3 provides background information on feature extraction techniques in the field of digital audio signal processing, which includes log spectral energies, mel-frequency cepstral coefficients (MFCCs), zero-crossing rate (ZCR) and kurtosis. This section also includes background information on different machine learning models that can be used to solve the classification problem. The text provides an introduction to logistic regression with regularization and the structure of convolutional neural networks (CNNs), explaining how CNNs work and how they are trained.
- Chapter 4 provides an analysis of the dataset used in this study. It describes the method of data collection and includes the rules that were used to classify the patients as TB positive or TB negative. A summary of statistics from the dataset is presented, as well as the division into training, development and test set partitions.
- Chapter 5 provides a broad overview of the experimental setup for the training and testing procedures. The different metrics used to evaluate the models are explained, including the confusion matrix, AUC and Youden Index. An overview of how the hyperparameters for each model were selected, along with the results from the experiments are discussed.
- Chapter 6 presents the experimental results of the test set and identifies the best model and its optimal features that should be used.
- Chapter 7 provides a discussion of the results in the context of this study and the broader field of cough classification. A section is also provided on further work that can be implemented and not performed in the time frame of this study.

# Chapter 2

## Literature Review

Previous studies have indicated that cough audio can be used to detect the presence of tuberculosis, COVID-19, and other respiratory diseases [2–4, 4–6]. The findings of each research study will be discussed for the potential of using cough sounds as a diagnostic tool and any insights provided into the field of cough audio analysis.

### **2.1. Detection of tuberculosis by automatic cough sound analysis [1]**

The dataset of this study consisted of only 38 patients (17 TB positive and 21 TB negative) and a total of 746 coughs between the two groups. The audio recordings of each patient are longer and consist of entire coughs and cough sequences. This differs from the fixed length audio recordings that will be used in this study. The findings of [1] show that a system designed using MFCCs was less successful than a system using log-spectral energies. It also found that improved performance was possible using filterbanks more closely spaced than those used to compute MFCCs. This implies that the system distinguishes differences in the audio that can not be heard by a human listener. The best system was a combination of features from the cough audio and clinical data obtained from the participants in which the decisions were combined using logistic regression, which obtained an AUC of 0.95. This study showed that on a small dataset, a good classifier performance can be achieved regarding TB detection in cough audio. The combination of metadata and cough audio yielded the best results, which shows the combination of the cough audio and the other data should be used to obtain a higher-performing classifier.

### **2.2. Automatic Cough Classification for Tuberculosis Screening in a Real-World Environment [2]**

The dataset of this study consisted of 16 patients who were confirmed TB positive and 35 patients who had respiratory problems which are indicative of TB but tested negative for TB. This is another smaller dataset similar to the one used in Botha *et al* [1]. The length



of each cough audio is similar to the data collected for this study. An extensive number of models were tested, including logistic regression, CNNs, KNNs, MLP, and support vector machines. The logistic regression model outperformed the other classifiers with an AUC of 0.86 and an accuracy of 84.54%. After performing sequential feature selection, the model improved to an AUC of 0.94. It was found that by using MFCCs without their first and second differentials, a near-optimal performance can be achieved. It also shows that MFCCs performed better than log spectral energies, which differs from the finding in Botha *et al* [1]. This could be due to differences in how the data was recorded or the small dataset that was used. Lastly, it was found that a close to optimal performance of the model can be found using as little as 4 features from the 78 features obtained during feature extraction.

## **2.3. Predicting Tuberculosis from Real-World Cough Audio Recordings and Metadata [3]**

This study uses the same dataset that we will use in the current study. Two different experiments were performed, one using only the cough data and another using a combination of metadata and cough audio. The study found in the cough-audio-only experiment an AUC of approximately 0.7 was obtained and by combining the metadata a higher AUC of approximately 0.81 was obtained. For the cough-only audio experiment, a convolutional neural network(CNN) was used. Cross-validation was used to determine the performance of the models. For the combined cough audio and metadata experiment, the fast logistic regression model performed almost as well as the complex models such as Random Forests and Adaptive Boosting. This study showed that a high AUC can be obtained from the given dataset even by using simple models such as logistic regression.

## **2.4. COVID-19 detection in cough, breath, and speech using deep transfer learning and bottleneck features [4]**

This study used a combination of different audio sources to identify COVID-19 in participants. Speech, cough, and breathing audio were utilized to extract features for COVID-19 classification. A CNN, Resnet50 and LSTM were trained using the data, and a close-to-optimal performance was obtained by the Resnet50 classifier with an AUC of 0.98, 0.94 and 0.92 for coughs, breaths and speech respectively. The features used in this study contained MFCCs, MFCCs with appended velocity, and MFCCs with appended velocity and acceleration with kurtosis and ZCR per frame of each audio file. Although pre-training

was performed on the models, this study shows that audio recordings of coughs, breaths and speech can all be used to detect the presence of COVID-19. The study also found that a higher number of MFCCs and a more densely populated filterbank performed better with the classifiers.

## **2.5. Development and clinical validation of Swaasa AI platform for screening and prioritization of pulmonary TB [5]**

This study published in September 2022 determined the feasibility of developing and using a platform to screen potential TB positive patients. The data was collected in India where 10-second cough audio recordings were taken from each participant and the symptoms were recorded and consisted of 195 TB positive participants and 152 TB negative participants. A total of 209 features were extracted, of which 170 features were used after correlation-based feature selection was used. The classifier consisted of a combined model which included a CNN which is trained with the mel spectrograms of the cough audio and an artificial neural network (ANN) for the other features. For the clinical validation of the model, it achieved an accuracy of 86.82% for determining if a participant is TB positive or TB negative with a sensitivity of 90.36% and specificity of 84.67%. This device was designed to be a screening tool and shows it can help in prioritizing and fast-tracking which participants should have to undergo a rapid test to confirm the diagnosis from the preliminary screening.

## **2.6. Deep Neural Network-Based Respiratory Pathology Classification Using Cough Sounds [6]**

This study published in August 2021 determined the feasibility of using deep-learning methods on cough audio to classify children's coughs as healthy, asthma, upper respiratory tract infection and lower respiratory tract infection. The data was collected on children with a clinical diagnosis and their cough sounds were recorded. The model used was a bidirectional long-short-term memory network using 14 MFCCs with their first and second differentials. This resulted in 42 coefficients for each frame of audio extracted. Five different models were designed. This included one that is used to classify between healthy coughs and pathological coughs, three others that classify between healthy coughs and each of the three pathological conditions and one model that classifies all four classes. The overall accuracy of the four-class model is higher than 84% which once again indicates the cough audio can be used to classify other respiratory diseases successfully and not only

TB.

## **2.7. Deep recurrent neural networks with attention mechanisms for respiratory anomaly classification [7]**

This study published in July 2021 used deep learning methods to create an architecture specifically used for audio-based classification on medical datasets. An attention bi-directional long-short-term memory network (A-BiLSTM) was used. The datasets used in this study contain the cough audio recordings of patients with different lung diseases (Asthma, Chronic Obstructive Pulmonary Disease (COPD), Bronchitis and Pneumonia) from the respiratory sound database as well as COVID-19 from the Cosara dataset. The feature extraction involved MFCCs with a frame size of 2048 samples and 75% frame overlap between successive frames. The best architecture obtains an accuracy of 96.2% for distinguishing between the different classes in the respiratory sound database and 96.8% for the Cosara dataset. This shows that the addition of the attention mechanism is useful for different cough-audio classification.

## **2.8. Conclusion**

In this chapter, multiple different studies were described and mentioned in which audio features were used to classify participants as TB positive or TB negative. The next chapter describes the different feature extraction techniques and an overview of the different models used in this study.

# Chapter 3

## Background

In this chapter, the different audio features that can be obtained from the cough audio, as well as the different machine learning models and techniques that were considered for solving the problem are described.

### 3.1. Features

Feature extraction is the process of computing a lower-dimensional representation of the raw input data that maintains the information important for the classification problem.

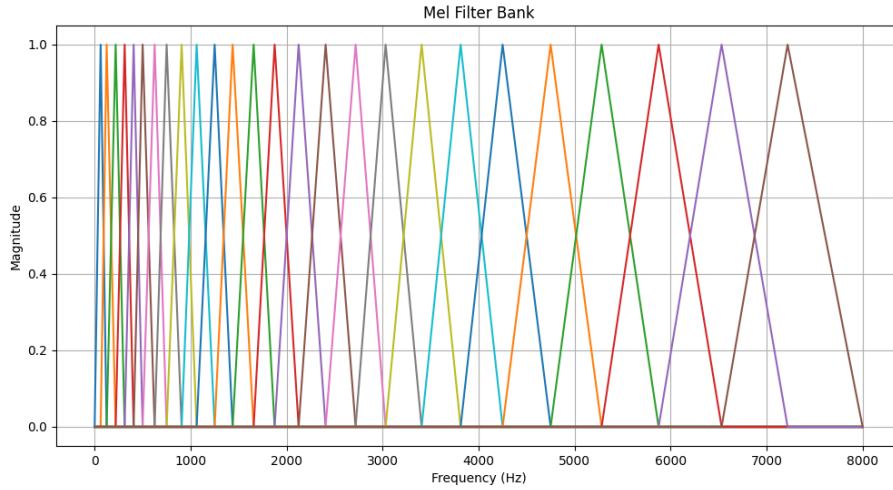
#### 3.1.1. Log Spectral Energies

Log spectral energies are a successful feature choice in audio analysis to identify tuberculosis and COVID-19 [1,2,4]. The filterbank used can be either linearly spaced triangular filters or mel-scale filters. The log spectral energies used in this study are calculated using a set of triangular filters which are linearly-spaced and overlapping. The process for calculating the log spectral energies is described below [1].

- The audio signal is separated into a sequence of successive and often overlapping frames and the fast Fourier transform is applied.
- A set of triangular filters that are linearly spaced and overlapping on the frequency scale and applied to the magnitude spectrum that was generated by the FFT.
- The logarithm of the filterbank energies is taken.
- The result of the logarithm on the filterbank energies is the log spectral energies. This will be a set of vectors that contains the energy for each frequency band of each frame.

#### 3.1.2. Mel-Frequency Cepstral Coefficients (MFCCs)

MFCCs have also been shown to be a successful feature choice in audio analysis in many applications, and be able to identify tuberculosis and COVID-19 from cough sounds [1,4,6].



**Figure 3.1:** Example of a logarithmically distributed triangular filterbank.

MFCCs use a non-linearly spaced filterbank as opposed to the linearly spaced filterbank described in Section 3.1.1, which mimics the frequency resolution of the human auditory system, to approximate the short-time signal spectrum. Computing MFCCs consists of a few steps as described in [2]. This is similar to computing the log spectral energies, except for using a set of filterbanks that are evenly spaced on the mel frequency scale and the discrete cosine transform (DCT) which is applied to the log spectral energies.

- The audio signal is separated into a sequence of successive and often overlapping frames and the fast fourier transform is applied to each.
- A number (typically between 20 and 40) of triangular filters distributed evenly on the mel frequency scale is determined and applied to the magnitude spectrum generated by the FFT.
- The logarithm of the filterbank energies is taken.
- The discrete cosine transform (DCT) is applied to the resulting log filterbank energies, resulting in the mel frequency cepstral coefficients (MFCCs).
- Additionally, the first and second temporal derivatives of the MFCCs can be computed and appended as features for subsequent audio analysis.

Figure 3.1 is an illustration of a mel-spaced filterbank with 26 filters. This shows how it mimics the human auditory system by capturing the non-linear frequency resolution of the hearing of a person.

### 3.1.3. Zero-crossing rate

The zero-crossing rate (ZCR) has been seen to have success as a feature of audio analysis [3, 4]. The zero-crossing rate (ZCR) is a measure of the number of times consecutive samples in a frame change signs [10]. The formula for calculating the average ZCR is shown in Equation 3.1:

$$ZCR = \frac{1}{N-1} \sum_{n=1}^{N-1} |\text{sign}(x[n]) - \text{sign}(x[n-1])| \quad (3.1)$$

where  $N$  is the number of samples in the frame,  $x[n]$  is the value of the signal at sample  $n$  and  $\text{sign}(x[n])$  returns the sign of  $x$  at sample  $n$  (either +1 or -1).

The zero-crossing rate is calculated per frame of the cough audio file, for which each audio file a vector of zero-crossing rates will be created.

### 3.1.4. Kurtosis

Kurtosis is a statistical measure that calculates the degree of peakedness or flatness of a probability distribution curve. It is calculated by standardizing the fourth moment of the population around the mean. Kurtosis is the measure of tailedness in a distribution where a positive kurtosis indicates peakness compared to the normal distribution, and a negative kurtosis has small tails and flatness [11]. Kurtosis is a successful feature in audio analysis [2, 3].

The formula for calculating population kurtosis is shown in equation 3.2 [3]:

$$\text{Kurtosis} = \frac{N(N+1)}{(N-1)(N-2)(N-3)} \left( \frac{\sum_{n=1}^N (x_n - \bar{x})^4}{\left( \sum_{n=1}^N (x_n - \bar{x})^2 \right)^2} \right) - \frac{3(N-1)^2}{(N-2)(N-3)} \quad (3.2)$$

or in simpler terms for sample kurtosis:

$$\text{Kurtosis} = \frac{1}{N-1} \sum_{n=1}^{N-1} \left( \frac{x_n - \mu}{\sigma} \right)^4 \quad (3.3)$$

where  $x_n$  is value of the sample at  $n$ ,  $\mu$  is the mean of the frame and  $\sigma$  is the standard deviation.

## 3.2. Models

Different model architectures can be implemented to build classifiers. The two different architectures explored included the baseline logistic regression model and convolutional neural networks.

### 3.2.1. Logistic Regression

According to Pahar *et al* [2] and in Botha *et al* [1], logistic regression has outperformed more sophisticated classifiers in certain clinical situations.

An LR model produces an output value between 0 and 1, which makes it useful in the case of the binary classification problem in this study. The idea behind logistic regression is to minimize the value of the negative log-likelihood, which is described in Equation 3.10. The quantity reflects the probability that a given input belongs to a certain class. The sigmoid function is at the core of logistic regression as this forces the input to a value between 0 and 1. The sigmoid function is given as follows:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (3.4)$$

where for logistic regression  $z$  is a linear combination of the input features and the weights associated with them:

$$\hat{y} = z = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \dots + \alpha_n x_n \quad (3.5)$$

In Equation 3.5  $\alpha$  represents the weights of each feature ( $\alpha_0$  being a scalar) and  $x_n$  representing the value of n-th feature. To determine which class is predicted by the logistic regression classifier, the value  $\hat{y}$  is interpreted as the probability that the input belongs to the class. Given a certain decision threshold  $p$ , the input is classified as positive (1) or negative (0) if:

$$\hat{y} = \begin{cases} 1 & \text{if } x \geq p \\ 0 & \text{if } x < p \end{cases} \quad (3.6)$$

Logistic regression is known to overfit when there are many features and too few samples. A penalty parameter can be added to the loss function to discourage large weight vectors that are often associated with overfitting. This is known as L2 regularisation. The penalty parameter added to the loss function is given as:

$$L2(\boldsymbol{\alpha}) = \lambda \|\boldsymbol{\alpha}\|_2^2 \quad (3.7)$$

In Equation 3.7 a regularisation parameter  $\lambda$  controls the strength of the regularization [12]. Larger values of  $\lambda$  lead to stronger regularization. The Euclidean norm of the weight vector is calculated, which will penalize larger weight values in the weight vector  $\boldsymbol{\alpha}$ .

### 3.2.2. Convolutional neural network (CNN)

Convolutional neural networks are a type of artificial neural network commonly used in processing image data. A basic CNN consists of three different layers, a convolutional

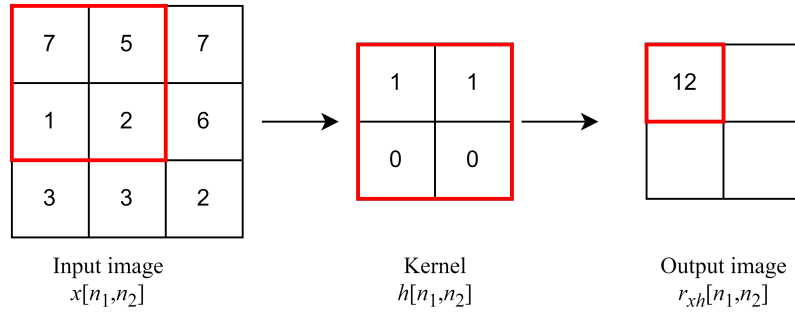
layer, a pooling layer and a fully connected layer [13]. The MFCCs or mel spectrograms extracted from the cough audio, can be used as input images to a CNN.

### Layers

The convolutional layer consists of kernels, also referred to as filters, which slide over the input image and perform convolution. Convolution is the mathematical operation that involves element-wise multiplication of the kernel and the overlapping area of the image and summing the results to obtain a single output. The formula for calculating the output of a kernel on an input feature map, is given as:

$$r_{xh}[n_1, n_2] = \sum_{i_1=n_1+1}^{n_1+N_{h1}} \sum_{i_2=n_2+1}^{n_2+N_{h2}} x[i_1, i_2] \cdot h(i_1 - n_1, i_2 - n_2) \quad (3.8)$$

where  $h(n_1, n_2)$  with dimensions  $(N_{h1} \times N_{h2})$  is the kernel image which is swept over the input image  $x[n_1, n_2]$  with dimensions  $(N_{x1} \times N_{x2})$  resulting in a new output image  $r_{xh}[n_1, n_2]$  with dimensions  $(N_{x1} - N_{h1} + 1) \times (N_{x2} - N_{h1} + 1)$  [14]. Figure 3.2 visually illustrates the application of the filter on the input image.



**Figure 3.2:** Visual illustration of applying a filter/kernel to an input image. This process is applied to each possible filter overlap on the input image resulting in a 2x2 output image. In this example  $N_{x1} = N_{x2} = 3$  and  $N_{h1} = N_{h2} = 2$ .

The single output is the input of an activation function, which for CNNs is usually the rectified linear unit function (ReLU). This function is defined as:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \quad (3.9)$$

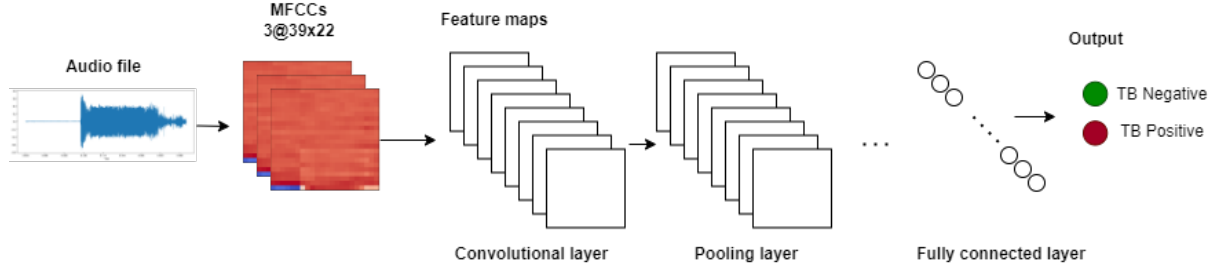
These kernels capture different features from the input image, such as edges, textures or even more complex patterns in the case of audio analysis.

The pooling layer reduces the output dimension of the convolutional layer. Pooling is done by taking another kernel, with no trainable parameters, and sweeping it over the output of the convolutional layer to reduce dimensionality. Two types of pooling have been



proposed, max pooling and average pooling. The most common max-pooling layer is a 2x2 kernel with a stride of 2 which scales the activation map down to 25% of its original size.

The final fully connected layer is used to make predictions based on the features extracted. Usually, the CNN output is rearranged into one vector which contains all the extracted features in a process which is called flattening. This vector is then the input to the fully connected layer or to a sigmoid function to perform the binary prediction.



**Figure 3.3:** The basic structure of a CNN which includes the 3-layered input image which contains the MFCCs, its first and its second differentials. It shows one convolutional layer with a pooling layer which can then be extended to multiple layers. The last layer is the fully connected layer which maps the output to two different classes for which a softmax activation function is applied to obtain the probabilities of the input belonging to a certain class. Adapted from [8].

## Gradient Descent

The convolutional neural network has many different parameters that need to be trained to be able to recognize features in the images. The loss function that is used to train the parameters is binary cross-entropy loss, which is a measure of the difference between two probability distributions. The CNN outputs a value between 0 and 1 which can be seen as the probability that each input belongs to the positive class. The formula for binary cross entropy loss is given as:

$$L(y, \hat{y}) = -(y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})) \quad (3.10)$$

where  $\hat{y}$  indicates the classifier output and  $y$  being the true Bernoulli distribution of the class labels. The optimal weights of the kernels are trained by minimizing the binary cross-entropy loss. This minimisation is achieved by performing gradient descent. The specific gradient descent algorithm used in this study is Adam [15]. The Adam optimizer is used since it is computationally efficient and requires little memory in comparison to other techniques [15]. The algorithm is described in Algorithm 3.1.

---

**Algorithm 3.1:** Adam optimizer for stochastic optimization. Adapted from [15].

---

Set hyperparameters:  $\alpha$  (learning rate),  $\beta_1$ ,  $\beta_2$  (Exponential decay rates for moment estimates), and  $\epsilon$ .

Let  $\theta_0$  be the initial parameter vector.

Set timestep  $t$  to 0. ( $t \leftarrow 0$ )

Initialize the 1st and 2nd moment vectors. ( $m_0 \leftarrow 0$ ,  $v_0 \leftarrow 0$ ).

**while**  $\theta_t$  not converged **do**

$t \leftarrow t + 1$

$g_t \leftarrow \nabla f(\theta_{t-1})$  (Get gradients with parameters at timestep  $t$ )

$m_t \leftarrow \beta_1 \times m_{t-1} + (1 - \beta_1) \times g_t$  (Update first moment estimate)

$v_t \leftarrow \beta_2 \times v_{t-1} + (1 - \beta_2) \times g_t^2$  (Update second moment estimate)

$\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  (Compute bias corrected first moment)

$\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  (Compute bias corrected second moment)

$\theta_t \leftarrow \theta_{t-1} - \alpha \times \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$  (Parameter vector update)

**end while**

**return**  $\theta_t$  (Resulting parameter vector)

---

### 3.3. Conclusion

This chapter provides a summary of the different audio features that have been successful in the field of audio classification. It also includes a discussion of the different types of classifiers which will be used for experimentation. The following chapter provides a discussion of the data collection procedure, a summary of the dataset statistics and the training, development and test partitions.

# Chapter 4

## Data

This chapter will describe the dataset that is used for experimentation in this project. A discussion of data collection, the complete dataset, and the division into training, development and testing partitions is provided.

### 4.1. Data collection procedure

The data used in this work originates from two different studies. The first is the R2D2 (Rapid Research in Diagnostics Development) for TB network, where the audio recordings were collected using the Hyfe Research app in South Africa, Uganda, India, Philippines, and Vietnam [3]. The second was carried out in Tanzania and Madagascar, and once again the audio was recorded using the Hyfe Research app [3, 16]. All patients who participated in these studies were over the age of 18 years and had visited a health centre due to health concerns and a new or worsening cough that had persisted for a minimum of two weeks [3]. The patients were instructed to follow the procedure required by the Hyfe app and to cough after a 3-2-1 countdown. The device then records for 5 seconds and an algorithm isolates the cough audio and captures half a second of audio in the identified time frame. A survey was also taken to gather personal and health-related information for every patient, which is given as metadata. The audio recordings were made at a sampling rate of 44100 Hz.

The complete dataset used consists of 1269 participants. Of these, 41 had no audio collected and were therefore disregarded. There were also 22 participants for which the audio files were not made available and these were also disregarded. The cough audio recordings were 0.5s long, with a mean cough audio recording length of 0.4989 seconds and a standard deviation of 0.0025. The number of such recordings per participant ranged between 1 and 50 with an average of 8.7. The lengths of the audio and the amount of each recording can be seen in Table 4.1. The table shows that more than 98% of the recordings are 0.5s in length.

Audio length (Seconds)	Number of recordings
0.34000	11
0.37873	12
0.38000	21
0.41873	37
0.42000	11
0.45873	24
0.46000	12
0.49873	25
0.50000	9451
<b>Total</b>	9604

**Table 4.1:** Audio recording lengths from participants.

## 4.2. Ground truth TB testing

Two different methods were used to determine whether a participant was TB positive or TB negative. The primary reference standard, labeled "Microbiological reference standard" in the metadata, included a set of rules for defining a patient as TB positive, TB negative or indeterminate. These rules were as follows:

TB-positive participants are patients for whom:

- A positive result was obtained for sputum Xpert (including grades very low, low, medium and high) OR
- A positive result was obtained for urine Xpert (including grades very low, low, medium and high) OR
- A positive culture was obtained in MGIT OR
- A positive culture was obtained in solid culture media (e.g., 7H10 Agar\*) OR
- A positive result was obtained on Xpert Ultra (including grades very low, low, medium and high) performed on a contaminated MGIT specimen OR
- Two trace Xpert Ultra results were positive on sputum and/or urine and/or contaminated MGIT

TB-negative participants are patients for whom:

- No positive result was obtained for sputum Xpert Ultra, urine Xpert Ultra, culture, Xpert Ultra on contaminated MGIT specimen AND
- At least one negative MGIT culture was obtained

Indeterminate results are patients for whom:

- No positive result (sputum Xpert Ultra, urine Xpert Ultra, culture, Xpert Ultra on contaminated MGIT specimen), AND
- < 2 trace Ultra results on sputum and/or urine and/or contaminated MGIT AND
- < 2 negative cultures due to contamination.

The Sputum Xpert reference standard is also used to determine patients as TB positive, TB negative or indeterminate. All participants received a sputum Xpert test and a repeat sputum Xpert test if the first test was indeterminate or positive-trace. The rules for this classification were defined as follows:

TB-positive participants are patients for whom:

- A positive baseline sputum Xpert result (including grades very low, low, medium and high), OR
- Two trace baseline sputum Xpert Ultra results

TB-negative participants will include patients with:

- A negative baseline sputum Xpert result, without any trace or positive baseline sputum Xpert Ultra results.

TB indeterminate patients included patients where neither criteria for TB positive nor for TB negative are met. Patients for whom tests did not return the same result, were classified as indeterminate.

The sputum Xpert test is a nucleic acid amplification test that works with the GeneXpert instrument system. A sputum sample is collected from the patient with possible TB. The sputum sample is mixed with a reagent and placed into the GeneXpert machine [17]. The urine Xpert test is similar but uses a urine sample instead of a sputum sample.

The Mycobacteria Growth Indicator Tube (MGIT) is another test used to detect TB. A 2-10ml saliva or sputum sample is taken and mixed with a reagent and placed in an incubator for an incubation period during which the instrument scans the sample regularly for increased fluorescence. Analysis of the fluorescence is used to determine if the sample is positive [18].

Solid media culture is the process of growing bacteria by adding a gelling agent such as agar with antimicrobial agents, to eliminate undesirable bacteria and to only select the bacteria desired to grow. A positive culture would indicate the presence of a bacterium called *Mycobacterium tuberculosis* [19].

### 4.3. Dataset summary

Table 4.2 summarises the dataset used in this study after cleaning has been performed. Cleaning included the removal of indeterminate cases, as well as the removal of participants who have cough audio missing or not available.

Country	TB-	TB+	Total	TB+ $\mu$	TB- $\mu$	TB+ $\sigma$	TB- $\sigma$	TB+ coughs	TB- coughs
India	125	14	139	0.5	0.4981	0	0.002	96	1035
Philippines	182	15	197	0.4999	0.4997	0	0.002	87	1264
Madagascar	74	60	134	0.5	0.4999	0	0.001	558	458
South Africa	112	24	136	0.4995	0.4995	0.001	0.001	224	894
Tanzania	80	9	89	0.4994	0.4977	0.003	0.007	104	987
Uganda	178	77	255	0.4997	0.4997	0.001	0.001	987	768
Vietnam	96	60	156	0.4965	0.4964	0.009	0.009	633	1066
<b>Total</b>	865	266	1106	0.4990	0.4989	0.0025	0.0025	2470	7134

**Table 4.2:** Table of TB participant data.

The columns "TB+" and "TB-" list the number of patients who were TB positive and negative respectively. The quantities  $\mu$  and  $\sigma$  are the mean and standard deviation of the cough audio samples. The final two columns indicate the number of coughs recorded.

In addition, metadata was collected for each participant as shown in Table 4.3.

### 4.4. Training, development and test set

The dataset was split into training, development and test sets by using a fixed 70/15/15 split ratio. The split was done per participant, to ensure there was no overlap of cough audio from the same participant between the different sets. The split was also done to ensure that a similar percentage of participants from each country occurred in each of the sets, while also balancing the number of TB-positive and TB-negative patients per country. Table 4.4 presents a summary of this data split.

### 4.5. Conclusion

This chapter provides a summary of how data was collected originally, how participants were classified as TB positive or negative, the complete dataset used, and the division of the training, development, and test sets. The following chapter will focus on the various experimental techniques utilized to evaluate the models, as well as the results obtained from these methods on the development set.

Feature	Description	Type
Sex	Male or female	Binary
Age	Age of the individual in years	Integer
Height	Height in centimeters	Integer
Weight	Weight in kilograms	Integer
HIV status	HIV infection status (positive, negative)	Binary
Duration of cough	Duration of cough in days	Integer
Prior TB	History of tuberculosis (yes, no, not sure)	Categorical
Prior TB type - Pulmonary	Asked at baseline: "With what kind of TB were you diagnosed?" (may select more than one) Participant selected pulmonary TB (Checked, Unchecked)	Binary
Prior TB type - Extrapulmonary	Asked at baseline: "With what kind of TB were you diagnosed?" (may select more than one) Participant selected extrapulmonary TB (Checked, Unchecked)	Binary
Prior TB type - Unknown	Asked at baseline: "With what kind of TB were you diagnosed?" (may select more than one) Participant selected Unknown (Checked, Unchecked)	Binary
Hemoptysis	Presence of coughing up blood (yes, no)	Binary
Fever	Presence of fever (yes, no)	Binary
Night sweats	Presence of night sweats (yes, no)	Binary
Heart rate	Heart rate in beats per minute	Integer
Temperature	Body temperature in degrees Celsius	Float
Weight loss	Weight loss in kilograms in the last 30 days (yes,no)	Binary
Smoked in the last week	Smoking behavior in the last week (yes, no)	Binary

**Table 4.3:** The metadata gathered for each participant in Table 4.2.

Country	Train Set			Development Set			Test Set		
	+ Patients	- Patients	%	+ Patients	- Patients	%	+ Patients	- Patients	%
	+ Coughs	- Coughs	%	+ Coughs	- Coughs	%	+ Coughs	- Coughs	%
India	10	88	10.2%	2	18	10%	2	19	9.5%
	67	711	8.6%	19	139	12.0%	10	185	5.1%
Madagascar	42	52	44.7%	9	11	45.0%	9	11	45.0%
	420	316	57.1%	85	77	52.5%	68	65	51.1%
Philippines	10	127	7.3%	2	28	6.7%	3	27	10.0%
	64	875	6.8%	7	200	3.4%	16	189	7.8%
South Africa	17	78	17.9%	4	17	19.0%	3	17	15.0%
	152	652	18.9%	52	119	30.4%	20	123	14.0%
Tanzania	6	56	9.7%	2	12	14.3%	1	12	7.7%
	75	670	10.1%	24	171	12.3%	5	146	3.3%
Uganda	54	125	30.1%	12	26	31.6%	11	27	28.9%
	472	985	32.4%	144	244	37.1%	152	201	43.1%
Vietnam	42	67	38.5%	9	14	39.1%	9	15	37.5%
	463	754	38.0%	74	164	31.1%	96	148	39.3%
Total	181	593	23.4%	40	126	24.1%	38	128	22.9%
	1713	4963	25.7%	405	1114	26.6%	367	1057	25.8%

**Table 4.4:** Table of the data distribution between the train, development and test splits. The first column of each set is the number of TB-positive patients and coughs per set. The second column of each set describes the number of TB-negative patients and coughs per set. The final % column is the percentage of positive patients or coughs per set.



# Chapter 5

## Experimental setup

This chapter will describe the different evaluation metrics used to evaluate the various models, the architecture used for experimentation, the features that were extracted and the hyperparameter tuning of the models.

### 5.1. Evaluation Metrics

It is important to evaluate each developed classifier with multiple metrics. All of the different evaluation metrics were implemented using the `scikit-learn` package in `Python`.

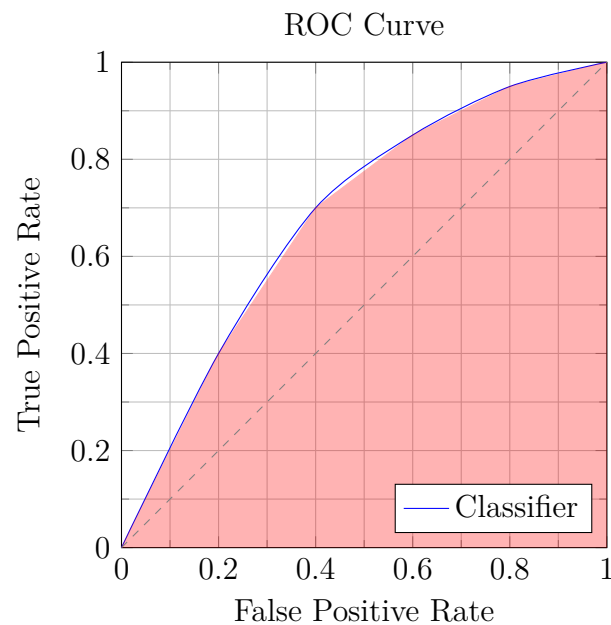
It is also important to clearly outline the outputs of each classifier in order to determine why certain evaluation metrics were used. The output of each classifier assigns a value  $p$  to each input (extracted cough audio features) where  $p \in [0, 1]$  indicating the probability the model assigns to the input cough originated from a TB positive patient. A decision threshold is applied to this probability to classify the cough as TB negative or TB positive. By comparing these classifier outputs with the ground truth labels, it can be determined how well the classifier performs with the chosen decision threshold by determining the confusion matrix. For binary classification, the confusion matrix consists of four counts: true positive (TP), true negative (TN), false positive (FP) and false negative (FN).

#### 5.1.1. Receiver Operating Characteristic (ROC)

The ROC curve is a visualization of the true positive rate (TPR) plotted on the y-axis and the false positive rate (FPR) plotted on the x-axis for different decision thresholds. The TPR is calculated as  $TPR = \frac{TP}{TP+FN}$  and  $FPR = \frac{FP}{FP+TN}$ . The receiver operating characteristic has been found to be a successful measure of accuracy for medical diagnostics, especially in the case of classifying patients as healthy or not healthy [20]. The curve is a plot of the sensitivity against 1-specificity, which is commonly used to analyse the performance of medical diagnostic models. Figure 5.2 is an example of an ROC curve, where the dotted line indicates the performance of a classifier making random decisions, and the blue line the performance of the model at different thresholds.

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

**Figure 5.1:** Confusion matrix illustrating the classification results with true positives (TP) and true negatives in green, false positives (FP) and false negative (FN) in red. The matrix depicts the relationship between actual positive and negative instances versus predicted positive and negative outcomes, providing insights into the performance of a classification model.



**Figure 5.2:** An example of a receiver operating characteristic (ROC) curve.

### 5.1.2. Area under the curve

The ROC curve is an excellent visualization of how the FPR and TPR compare at different decision thresholds, but it is sometimes difficult to compare two different models based on the curve alone. The area under the curve (AUC) is often used in preference as a "single number" evaluation of machine learning algorithms compared to other performance measures [21]. The AUC is  $\in [0, 1]$ , where values closer to 1 indicate a perfect classifier, and the value of 0.5 indicates a classifier making random decisions. The higher the value of the AUC, the greater the performance of the model. The shaded area in Figure 5.2 shows the area used for the calculation of AUC.

### 5.1.3. Youden Index

The Youden Index is a statistic used to evaluate the performance of a binary classification model [22]. It is the threshold that optimizes the differentiating ability of a model by choosing a threshold that maximizes the sum of the sensitivity and specificity. A higher Youden Index indicates a better discrimination ability and a higher performance of the model [22]. The Youden index is calculated as:

$$J = \text{Sensitivity} + \text{Specificity} - 1 = \frac{\text{TPR} + \text{TNR} - 1}{2} \quad (5.1)$$

### 5.1.4. Per-cough and per-participant assessment

Two different methods of assessment will be used for the models to determine their performance. The first method involves per-cough assessment, where each separate cough is classified individually, and no reference is made to the participant. The second method involves per-participant assessment, where the classifier output for all coughs belonging to a certain participant is aggregated to produce new combined output. This per-participant assessment is also used for the inclusion of metadata as mentioned in Section 5.2.3.

### 5.1.5. Software tools

The work done for experimentation is done in `Python v3.10`. The Python library used for feature extraction is `Librosa v0.10.1`, the logistic regression models and convolutional neural networks built with `Pytorch v2.0.1`. The experiments were run on two different machines, one with an Intel i7-10th Gen CPU and on a shared computer from the Digital Signal Processing lab which used `Cuda` to train the models at a faster rate.

## 5.2. Feature Extraction

From the cough-audio, features were extracted which formed the input to the different models. The audio features include MFCCs and mel spectrograms. In addition, non-audio features were available in the form of metadata that was obtained from the participant surveys. All the audio features were normalized before training, and there was no overlap of participants between the train, development and test sets to prevent overtraining.

### 5.2.1. Framing

A frame is a short fixed-length sub-portion of the audio signal, from which features will be extracted. Frames can overlap to increase the time resolution of feature extraction. The cough audio was sampled at 44.1kHz and a fixed frame size of 1024 samples with an overlap of 75%, 50%, 25%, and 0% per frame were tested. The frame size and overlap were chosen based on the results found in [1] leaving some room for experimentation. The number of frames is calculated as:

$$\# \text{ of frames} = \left\lfloor \frac{\text{Signal Duration (in seconds)}}{\text{Frame Duration (in seconds)} - \text{Frame Overlap (in seconds)}} \right\rfloor + 1 \quad (5.2)$$

which for example gives 22 frames for an audio duration of 500ms and 0% overlap.

The window type is the Hanning window, in which the positive cosine function is applied to each frame. The percentage overlap between successive frames was a hyperparameter that was tuned during the training process.

Percentage overlap (%)	Number of frames
0	22
25	29
50	44
75	87

**Table 5.1:** Table of the number of frames from the audio for each different % overlap between successive frames.

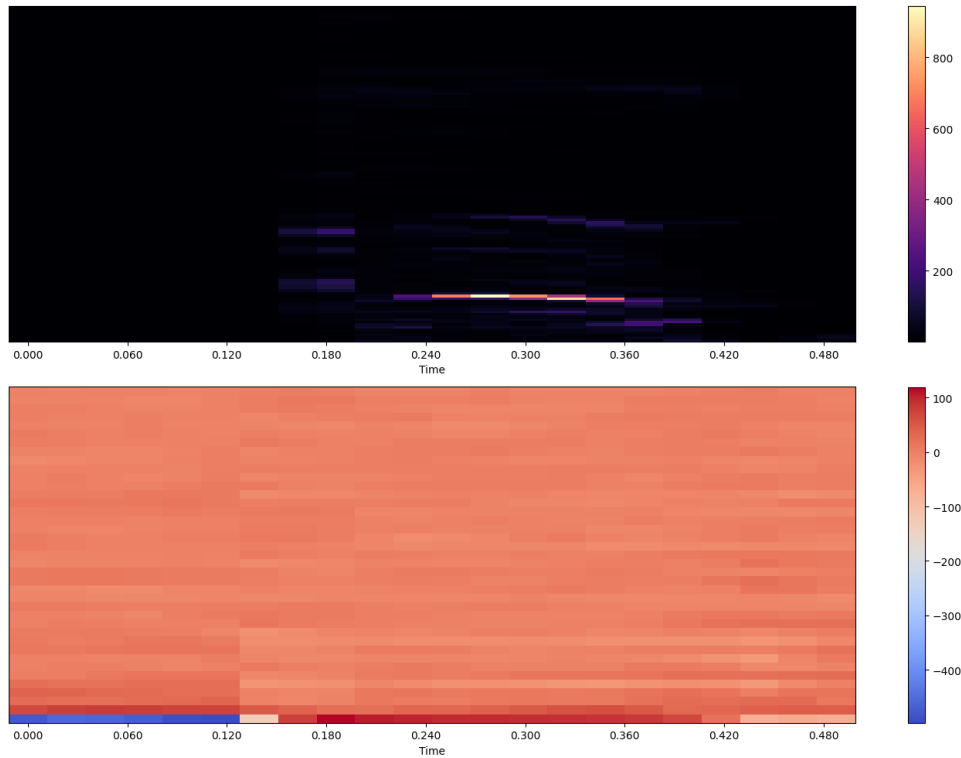
### 5.2.2. MFCCs and mel spectrogram

The MFCCs are features used in logistic regression and CNN models. The number of coefficients varied from 13, 26 and 39 during training and validation. The number of channels for the mel spectrogram was chosen as 128 was also used in [3]. Figure 5.3 shows an example mel spectrogram and MFCCs with 39 coefficients for a TB-positive cough. It was found from preliminary testing that MFCCs outperformed the mel spectrograms when used as features. The number of cepstral coefficients to include in the MFCC feature

vector is a hyperparameter that was explored during training and validation to determine if a larger number of coefficients leads to increased performance. The first and second differentials of the MFCCs were also extracted and used as additional input to the different models. For logistic regression, the initial experiments showed no performance increase when these differentials were added as input to the models, but for the CNNs it increased performance considerably and was used for further experimentation.

### 5.2.3. Metadata

The metadata that was collected for each participant will also be used in our experiments. A list of these metadata can be found in Table 4.3. The categorical columns were transformed to a one-hot encoding in order to be suitable as input for logistic regression models. The metadata is included to determine if a better-performing classifier can be developed by presenting the metadata and the output of the cough-audio classifier to a logistic regression model. The metadata will serve as additional features that will be combined with the audio features in an attempt to improve the performance of the classifier as if it were a real-world scenario.



**Figure 5.3:** A mel spectrogram with 128 filters and MFCCs with 39 coefficients extracted from a positive TB cough.

### 5.2.4. Feature extraction hyperparameters

The different feature extraction hyperparameters which needed to be optimised include the frame overlap ( $\delta$ ) and the number of MFCCs ( $\mathcal{M}$ ). Table 5.2 displays the two hyperparameters for feature extraction and their corresponding value ranges.

Hyperparameter	Description	Range
Frame overlap ( $\delta$ )	The % overlap between successive frames of the cough audio	$\delta=0\%$ , 25%, 50% and 75%
MFCCs ( $\mathcal{M}$ )	The number of MFCCs	$\mathcal{M}=13$ , 26 and 39

**Table 5.2:** Feature extraction hyperparameters and the ranges considered.

## 5.3. Hyperparameter tuning

The choice of hyperparameters influences the training and ultimately the performance of the models. These hyperparameters are tuned by observing the performance of the model on the development set. The best choice of hyperparameters based on the performance of the development set will then be used on the held-out test set to evaluate the performance of the model.

### 5.3.1. Logistic Regression

As a baseline model with which to compare the performance of the more complex CNNs, a logistic regression model is trained and evaluated using the different extracted features. The development set is used to evaluate the performance of different hyperparameters for logistic regression. The logistic regression model was implemented using `Pytorch`, audio feature extraction was accomplished using `librosa`, and the evaluation metrics and ROC curves were calculated using `sklearn`. The feature extraction has many hyperparameters that can be optimised to find the optimal performing logistic regression model. For MFCCs and mel spectrograms, the features extracted from all the frames in the audio signal were flattened for input to a logistic regression model. Different  $\lambda$  values for L2 regularisation will be used to reduce overfitting.

For logistic regression, the hyperparameters that needed to be optimised were the L2 regularisation strength ( $\lambda$ ) and the learning rate ( $\eta$ ). Table 5.3 lists the different hyperparameters which were optimised during training for logistic regression.

Previous studies have used different frame overlap percentages of frames with success, where [3] used a 50% overlap between frames while others used no overlap between

Hyperparameter	Description	Range
L2 regularisation strength ( $\lambda$ )	The strength of the regularisation which discourages large singular weight values	$\lambda=10^i$ where $i=-6,-5,\dots,-1,0$
Learning rate ( $\eta$ )	The rate at which the model learns. Too small values lead to slow convergence and large values can lead to overshooting and not converging	$\eta=10^i$ where $i=-5,-4$

**Table 5.3:** Different hyperparameters for logistic regression and the range on what optimisation techniques were performed.

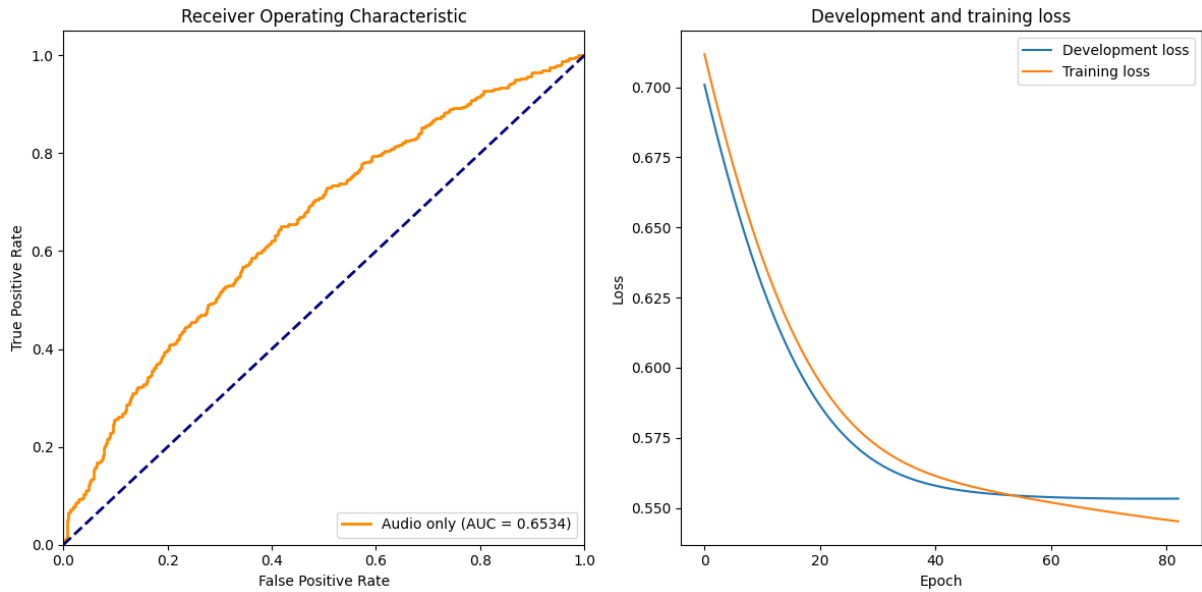
frames [1]. The choice of a fixed frame length of 1024 samples (corresponding to 23ms) was taken as it was done in [1]. For each different combination of feature extraction hyperparameters, the L2 regularisation parameter  $\lambda$  was varied to see the effect of larger regularisation. The loss function chosen is the binary cross entropy. Adam is chosen as an optimizer with the parameters  $\beta_1 = 0.9$ ,  $\beta_2 = 0.99$  and  $\epsilon = 10^{-8}$  as recommended in [15]. Early stopping was implemented to reduce the chances of overfitting. The loss on the development set is monitored and if the loss on the development set does not decrease for 10 epochs, training is stopped. Logistic regression is a model which is easily trained and does not easily overfit the data.

**Table 5.4:** Logistic regression hyperparameter tuning results. The best result for each combination of % overlap, number of MFCCs and  $\lambda$  are shown in the table based on the performance on the development set.

Frame Length (Samples)	Overlap ( $\delta$ )	MFCCs ( $\mathcal{M}$ )	AUC	L2 Regularization Loss ( $\lambda$ )
1024	0%	13	0.6460	0.01
1024	0%	26	0.6523	0.1
1024	0%	39	0.6503	0.1
1024	25%	13	<b>0.6534</b>	0.001
1024	25%	26	0.6485	0.1
1024	25%	39	0.6444	0.1
1024	50%	13	0.6510	0.001
1024	50%	26	0.6493	0.1
1024	50%	39	0.6462	0.1
1024	75%	13	0.6451	0.0001
1024	75%	26	0.6361	0.1
1024	75%	39	0.6328	0.1

From the grid search, the best-performing model on the development set using MFCCs has 13 coefficients and a regularisation loss parameter ( $\lambda$ ) of 0.001. An interesting point

to note is that the model performed better on average with fewer MFCCs. This could be due to a large input dimensionality that results when flattening the MFCCs as is required for a logistic regression model. The best-performing model is then used to classify development-set participants as described in Section 5.1.4, after which the ROC curve is plotted. By scoring in this way, the performance of the model increased to an AUC of 0.6769 and a Youden index of 0.2666 on the development set. The ROC curve, the development and the training loss are shown in Figure 5.4. This figure clearly shows that the model trains correctly as the loss on the development and training set decreases for each epoch. The output of this logistic regression model was then added as a feature into a second logistic regression model which included the metadata. The metadata features were adjusted to be used with a logistic regression model as described in Section 5.2.3. The model is then trained again and the AUC of the model improves to 0.7927 and a Youden index of 0.4333. Figure 5.5 displays the AUC curves for the cough-audio-only and cough audio with metadata experiments on the development set. The results show that the addition of metadata improves the model's classification performance considerably.

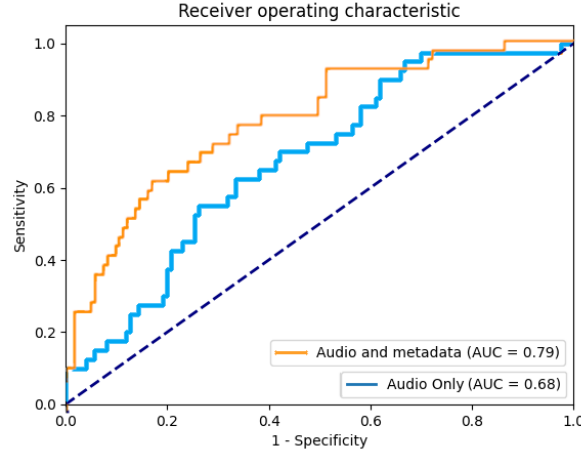


**Figure 5.4:** The development set ROC curve of the top-performing audio-only logistic regression model using 13 MFCCs extracted from 1024-point frames that overlap by 25% is shown on the left. On the right, the development and training losses are plotted as the model was trained. The point where the development loss levels off indicates the start of overfitting, which is why early stopping was implemented.

### 5.3.2. Convolutional Neural Network

Convolutional neural networks (CNNs) are trained and compared to the baseline logistic regression model in a cough-audio only experiment, as a CNN is suited to handle image input but not metadata. A CNN has many different architectural hyperparameters that





**Figure 5.5:** ROC curves of the participant which compares the cough-only experiment with the cough and metadata experiment.

can be optimised, for example, the number of convolutional layers ( $l$ ), dropout layers with different dropout rates ( $\mathbb{D}$ ), the learning rate ( $\eta$ ) and the batch size ( $B$ ) used during training. Some hyperparameters were fixed due to time constraints since training was time-consuming. The fixed hyperparameters were chosen based on a previous study [3]. Table 5.5 indicates which hyperparameters were fixed and which were optimised in the development process.

The fixed hyperparameters were chosen to reduce the time it would take to perform hyperparameter optimisation. A larger number of convolutional layers ( $l$ ) leads to a more complex model with more parameters to train. The learning rate influences how fast the model learns, and is often seen as a key hyperparameter [23]. Early stopping is once again implemented to prevent overfitting with patience of 10. The batch size is the number of samples processed in a single forward and backward pass before a weight update is performed and one epoch will be completed once all the batches have been passed through the network. The dropout layer forces certain inputs to 0 with a probability specified as the dropout rate. The purpose of the dropout layer is to reduce overfitting of the convolutional neural network. As a final layer to the CNN, a sigmoid activation function is used to force the output to a value between 0 or 1. The loss function used is binary cross entropy loss. Similarly to the logistic regression model, Adam is chosen as an optimizer with the recommended parameters  $\beta_1 = 0.9$ ,  $\beta_2 = 0.99$  and  $\epsilon = 10^{-8}$  as recommended in [15].

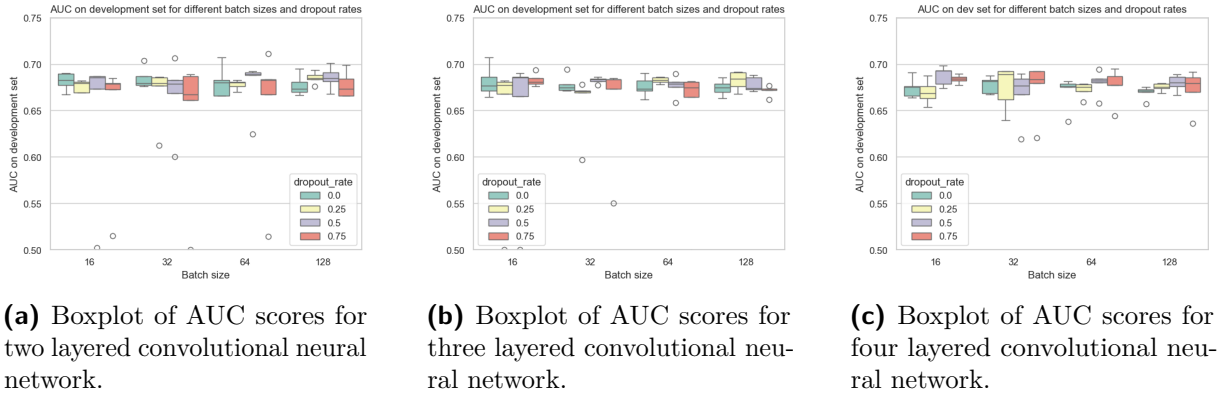
### Three layered MFCCs input

During the initial experimentation, it was found that a larger number of MFCCs led to a significantly increased performance on the development set. For this reason, 39 MFCCs were used for the grid search in contrast to the 13 MFCCs which led to the best model

**Table 5.5:** Table of the different constant and tuned hyperparameters used in the experimental process for CNNs. A short description as well as the range of the different hyperparameters are listed.

Parameter	Description	Range
<b>Fixed Parameters</b>		
Number of kernels	The number of filters in each convolutional layer	[16,32],[16,32,64] and [16,32,64,128] for the 2,3 and 4 layered CNNs.
Kernel size	The size of the kernel at each convolutional layer	3x3 with a padding of 1
Fully connected layers	The number of fully connected layers after the convolutional layers	3 layers (512, 256, 128 neurons in each layer)
Activation function	The activation function which is used after each convolutional layer and fully connected layer	ReLU
Pooling	Type of pooling used in the experiments	Max pooling (Kernel size of 2x2 and stride of 2)
Batch normalization	Batch normalization performed after each convolutional output	Yes
<b>Tuned Parameters</b>		
Number of Layers	The number of convolutional and pooling layers	$l=2, 3$ and $4$
Learning Rate	The learning rate for the optimizer	$\eta=10^i$ where $i=-7, -6, \dots -2, -1$
Batch Size	The number of samples in each batch	$B=16k$ where $k=1,2,3,4$
Dropout Rate	The dropout rate to prevent overfitting	$\mathbb{D}=0.25k$ where $k=0,1,2,3$

for logistic regression. The model input image consisted of three layers: one for MFCCs, one for its first differential, and a final layer for its second differentials. Three-layered images are commonly used for CNNs, where each layer accounts for a different RGB colour channel. In order to analyze the results, any models that had an AUC of 0.5 or less were removed from the analysis. The reason behind this is that in some cases too large or too small learning rates caused the model to learn too slowly or not converge at all. For each hyperparameter optimisation, a predetermined random seed was specified to initialize the model's weights and the batch shuffling process during training. This ensured that each model was trained with the same initial conditions to find the best architecture. Figure 5.8 displays a boxplot of the different AUC scores achieved by the different models with different hyperparameters.

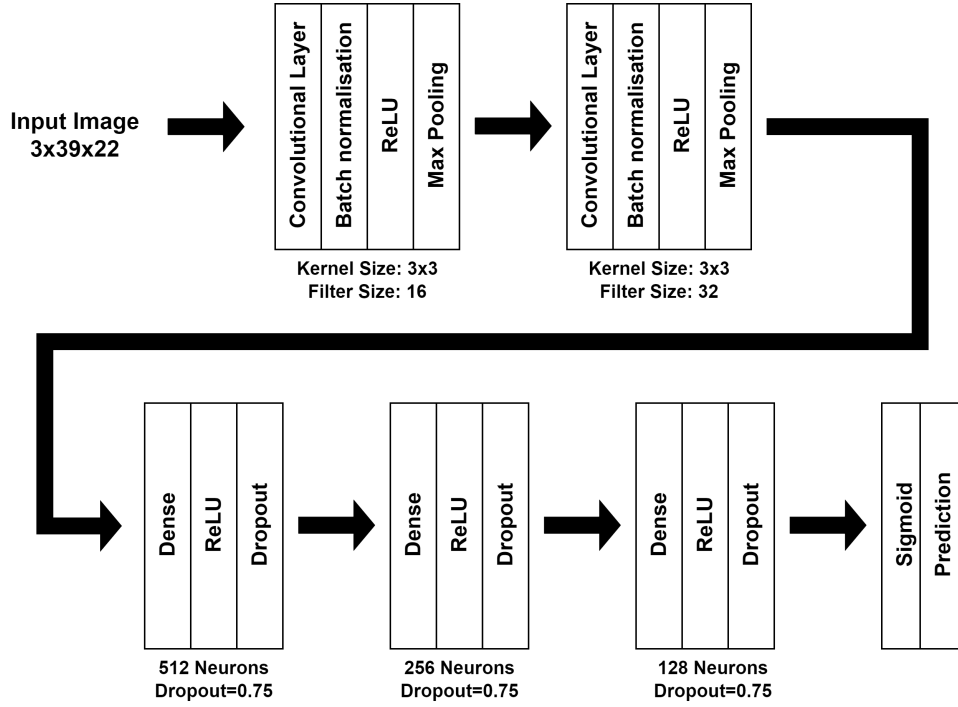


**Figure 5.6:** AUC scores for convolutional neural networks trained with two, three, and four convolutional layers showed varying AUC distributions. Larger CNNs had a higher variance due to the larger number of trainable parameters.

The best performance obtained from the grid search was for a two-layered ( $l = 2$ ) CNN, a batch size of 64 ( $B = 64$ ), dropout rate of 0.75 ( $\mathbb{D}=0.75$ ) and a learning rate of  $10^{-3}$  ( $\eta = 10^{-3}$ ). This CNN classifier achieved an AUC of 0.7110 and a Youden's index of 0.3327 for cough classification on the development set. As for the logistic regression model, this CNN was also used to classify participants as TB positive and TB negative, resulting in an AUC of 0.7167 and a best Youden index of 0.3726. Figure 5.7 displays the best CNN architecture found from the gridsearch. With the addition of metadata with the classifier output, the model achieved an AUC of 0.7962 and a Youden Index of 0.4552, which is significantly better.

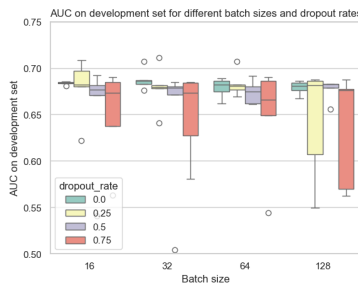
### Combined feature matrix

As a second CNN experimental setup, a different feature matrix was considered as input. The MFCCs, together with their first and second differentials, and the values per frame of the zero-crossing rate and kurtosis were combined to create a new feature matrix. This feature matrix was a one-layered input since the differentials are now stacked and not given

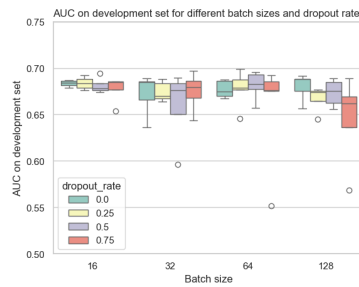


**Figure 5.7:** Architecture of the best performing CNN model with the optimal hyperparameters tuned on the development set. It shows the different convolutional and fully connected layers with the dropout rate used.

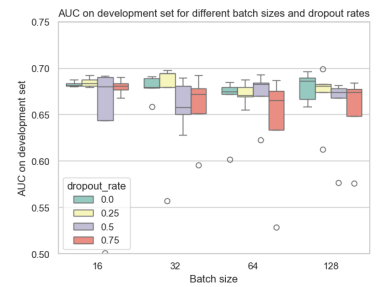
as three different input channels to the CNN. This was done to investigate if the addition of kurtosis and zero-crossing rate had an effect on the performance of the models. As for the three-layered MFCC input, for each hyperparameter optimisation, a predetermined random seed was specified to initialize the model's weights and the batch shuffling process during training. This ensured that each model was trained with the same initial conditions to find the best architecture.



**(a)** Boxplot of AUC scores for two layered convolutional neural network.



**(b)** Boxplot of AUC scores for three layered convolutional neural network.



**(c)** Boxplot of AUC scores for four layered convolutional neural network.

**Figure 5.8:** AUC scores for convolutional neural networks trained with two, three, and four convolutional layers showed varying AUC distributions. Larger CNNs had a higher variance due to the larger number of trainable parameters.

For large batch sizes the models showed a large variance in performances across the 2,3 and 4 convolutional layered models. The best hyperparameters obtained from the

grid search were two layers ( $l = 2$ ), a batch size of 32 ( $B = 32$ ), dropout rate of 0.25 ( $\mathbb{D}=0.25$ ) and a learning rate of  $10^{-3}$  ( $\eta = 10^{-3}$ ). The model had an AUC of 0.7113 and a Youden index of 0.3444. The per-participant assessment led to an AUC of 0.7242 and a Youden index of 0.377. This shows that the inclusion of ZCR and kurtosis showed a slightly improved performance on the development set for audio-only cough classification and performed better for the audio-only participant classification than the three-layered MFCCs.

## 5.4. Conclusion

This chapter provides a description of what evaluation metrics were used, how the experiments were performed, and what features were extracted from the cough audio and metadata. It concludes with a discussion on hyperparameter tuning, presenting the results of the best hyperparameters and the performance on the development set. The optimal hyperparameters for the models are listed in Table 5.6. The next chapter discusses the performance of the models on the hold-out test set.

Model	Best hyperparameters
Logistic Regression	L2 Regularisation Loss ( $\lambda=0.001$ ) and Learning Rate ( $\eta=10^{-4}$ )
CNN with three layered MFCCs	Layers ( $l = 2$ ), Batch size ( $B = 64$ ), Dropout rate ( $\mathbb{D}=0.75$ ) and learning rate $10^{-3}$ ( $\eta = 10^{-3}$ ).
CNN with ZCR and kurtosis included	Layers ( $l = 2$ ), Batch size ( $B = 32$ ), Dropout rate ( $\mathbb{D}=0.25$ ) and learning rate $10^{-3}$ ( $\eta = 10^{-3}$ ).

**Table 5.6:** Table of the best hyperparameters found during the grid search for each of the different models as measured on the development set.

# Chapter 6

## Experimental Results

In this chapter, the previously mentioned best models are evaluated on the held-out test set in terms of AUC, sensitivity and specificity and the results discussed. The models use the specific hyperparameters found in Section 5.3, but are retrained on the combined train set and development set. The test set was held out to ensure independence from the training and development data.

### 6.1. Logistic Regression

The final results of the logistic regression model for cough-only classification are shown in Table 6.1. For per-cough audio-only, the model achieved an AUC of 0.7020 on the test set and per-participant audio-only classification using only the cough features achieved an AUC of 0.7060. By including the metadata, the per-participant performance increased to an AUC of 0.8396.

### 6.2. Convolutional Neural Network

The final results for the convolutional neural network for per cough audio-only classification are shown in Table 6.2a. The test set for per-cough audio-only classification achieved an AUC of 0.7709 with the three-layered MFCCs input model compared to the model which included the kurtosis and the zero-crossing rate for per-cough audio-only classification which achieved an AUC of 0.7468.

Similarly, the final results for the convolutional neural network for per-participant audio-only classification are shown in Table 6.2b. The test set for per-participant audio-only achieved an AUC of 0.7410 with the three-layered MFCCs input model compared to the model which included the kurtosis and the zero-crossing rate for per-participant audio-only which achieved an AUC of 0.7105 which is a decrease from the per-cough audio-only classification.

The performance again increased with the inclusion of metadata to achieve an AUC of 0.8380 for per-participant classification with the three-layered MFCCs input and AUC of 0.8353 for per-participant with the model which included kurtosis and the zero-crossing

**(a)** Results of the logistic regression classifier for audio-only cough classification.

Dataset	AUC	Sensitivity (True Positive Rate)	Specificity (True Negative Rate)
Development	0.6534	0.5835	0.6475
Test	0.7020	0.7493	0.5790

**(b)** Results of the logistic regression classifier for audio-only participant classification.

Dataset	AUC	Sensitivity (True Positive Rate)	Specificity (True Negative Rate)
Development	0.6769	0.6	0.6666
Test	0.7060	0.8158	0.5938

**(c)** Results of the logistic regression classifier audio and metadata participant classification.

Dataset	AUC	Sensitivity (True Positive Rate)	Specificity (True Negative Rate)
Development	0.7927	0.6	0.8333
Test	0.8396	0.7368	0.8438

**Table 6.1:** Performance of the logistic regression models with optimised hyperparameters on the development and on the test set. Results are shown for (a) audio-only cough classification (b) audio-only participant classification and (c) audio and metadata per-participant classification.

rate. These results are shown in Table 6.2c.

### 6.3. Summary

As a summary of the different results, Figure 6.1 displays two different graphs to show the performance of the model on the test data. The best-performing model on cough-only classification with an AUC of 0.7709 was the optimized CNN model which takes three layered MFCCs as input. The best-performing model for per-participant classification was the metadata included with the output of the logistic regression model, although both the optimized CNNs had very similar AUC scores. This shows that the metadata has features that help the classification performance of the model which is not possible with the cough-audio alone. The CNN models performed better on the hold-out test set for cough classification, for both input features used. It was also found that the more complex CNNs performed better than the logistic regression models for the audio-only per-cough and per-participant classification for audio-only results on the test set. However, the gap between the logistic regression and CNN was closed when the metadata was added where similar results were found.

**(a)** Results of the CNN classifiers for audio-only cough classification.

<b>Dataset</b>	<b>AUC</b>	<b>Sensitivity (True Positive Rate)</b>	<b>Specificity (True Negative Rate)</b>
Development	0.7110	0.7025	0.6302
Test	0.7709	0.7766	0.6641
Development + ZCR + Kurtosis	0.7113	0.8175	0.5269
Test + ZCR + Kurtosis	0.7468	0.7684	0.6575

**(b)** Results of the CNN classifiers for audio-only participant classification.

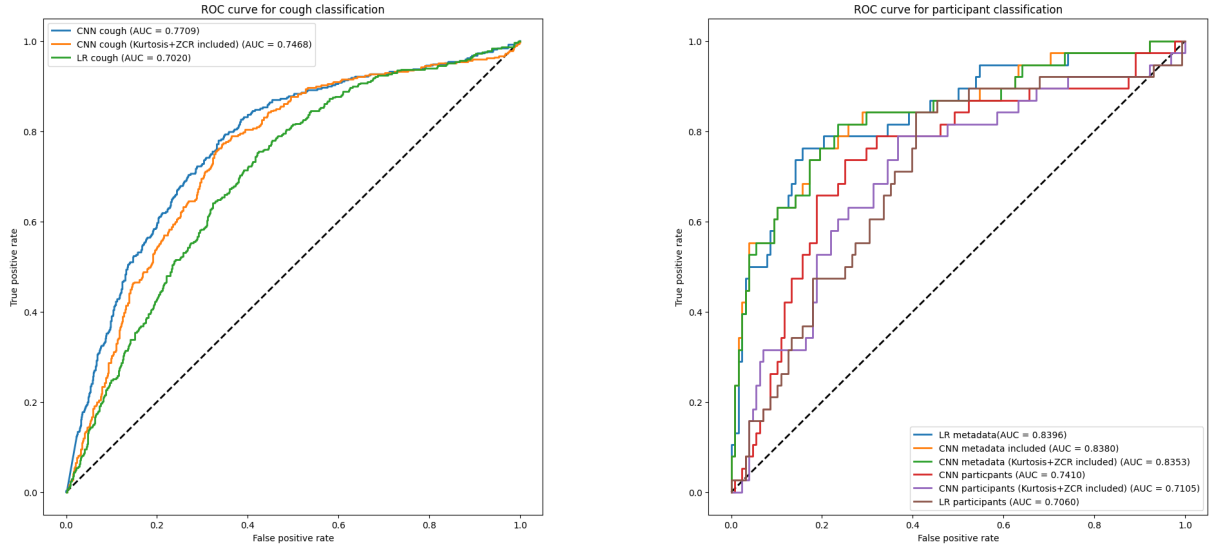
<b>Dataset</b>	<b>AUC</b>	<b>Sensitivity (True Positive Rate)</b>	<b>Specificity (True Negative Rate)</b>
Development	0.7167	0.825	0.5476
Test	0.7410	0.7105	0.75
Development + ZCR + Kurtosis	0.7242	0.75	0.627
Test + ZCR + Kurtosis	0.7105	0.7632	0.6328

**(c)** Results of the CNN classifiers for audio and metadata participant classification.

<b>Dataset</b>	<b>AUC</b>	<b>Sensitivity (True Positive Rate)</b>	<b>Specificity (True Negative Rate)</b>
Development	0.7962	0.725	0.7302
Test	0.8380	0.7368	0.8047
Development + ZCR + Kurtosis	0.8014	0.725	0.7381
Test + ZCR + Kurtosis	0.8353	0.7895	0.7656

**Table 6.2:** Performance of the CNN models with optimised hyperparameters on the development and on the test set. Results are shown for (a) audio-only cough classification (b) audio-only participant classification and (c) audio and metadata per-participant classification





**(a)** ROC curves of the best-performing models on the test set for per-cough classification.

**(b)** ROC curves of the best-performing models on the test set for per-participant classification.

**Figure 6.1:** Test set ROC curves for per-cough and per-participant classification. For per-cough classification, the best performing model was the two convolutional-layered CNN using the three-layered MFCCs as input. For per-participant classification, the best model was the metadata combined with the logistic regression results.

## 6.4. Conclusion

This chapter discusses the results of the different best-performing models on the hold-out test set, which includes the logistic regression classifier and the convolutional neural networks. The differences in the performance of the models are discussed as well as the effect of inclusion of the metadata on performance. Table 6.3 shows the best-performing model on the test set for each of the different classification methods used. The following chapter will provide a short conclusion and summary of the project, as well as any further work that is suggested for continuation.

Classification type	Classifier	AUC on test set
Audio-only per-cough classification	CNN with three-layered MFCCs as input	0.771
Audio-only per-participant classification	CNN with three-layered MFCCs as input	0.741
Audio and metadata per-participant classification	Logistic Regression with audio and metadata input	0.839

**Table 6.3:** Best classifier for the audio-only per-cough classification, audio-only per-participant classification and audio and metadata per-participant classification.

# Chapter 7

## Discussion and Conclusion

This project aimed to determine the feasibility of using short cough audio recordings from participants to build classifiers that classify coughs and participants as TB positive or TB negative. Different features were extracted from the cough audio recordings, for which different model architectures were designed and trained, and then tested on unseen participants. The results of a survey of participants, which included clinical data collected at the date of recording, was added to assess if an improvement could be made to the model performance.

### 7.1. Main findings

Audio features were extracted from the cough audio of each participant, which included both MFCCs and mel spectrograms during early testing. MFCCs showed more promising results, with associated feature extraction hyperparameters including frame overlap and the number of MFCCs. Two different classifier architectures were used to develop many different classifiers and find their optimal hyperparameters through a grid search. The classifier and the hyperparameters were evaluated on the development set by comparing the AUC scores. Metadata was included in the best classifier output of each architecture and shown to improve classifier performance. The best architectures had an AUC score of above 0.83 for per-participant classification with metadata included, and an AUC score above 0.77 when classifying coughs based only on the audio.

This project shows that there is potential for using short cough-audio recordings from participants to aid in diagnosing tuberculosis. The more complex convolutional neural networks outperformed the logistic regression classifiers on the test set for per cough classification and per participant classification based only on the input when audio features were combined with clinical metadata, the performance increased considerably for both architectures. The results of this study are consistent with a previous study that used the same dataset but a different training, development, and test split. Similar AUC scores were obtained, even though different frame sizes were used and the audio was downsampled [3]. This shows that a system in which cough audio and clinical data are recorded can be used to help diagnose patients with tuberculosis is possible.

## 7.2. Future Work

Future work on the problem of TB diagnosis will benefit from even more data and different architectures that can be explored. In terms of the current architectures used, a non-exhaustive search was performed in terms of parameter optimisation for the convolutional neural networks. The number of kernels and their sizes can be varied as well as different activation functions and types of pooling. In terms of hyperparameters for feature extraction, different frame sizes and numbers of MFCCs can still be tested. Data augmentation techniques can also be applied to the audio, which includes pitch-shifting or time-stretching to diversify the training set. There are also many different architectures that can still be explored for TB diagnosis. A recurrent neural network (RNN) can be tested which will help capture the temporal dependencies in the audio as well as different ResNets architectures can be considered.

# Bibliography

- [1] G. H. R. Botha, G. Theron, R. M. Warren, M. Klopper, K. Dheda, P. D. van Helden, and T. R. Niesler, “Detection of tuberculosis by automatic cough sound analysis,” *Physiological Measurement*, vol. 39, no. 4, p. 045005, apr 2018.
- [2] M. Pahar, M. Klopper, B. Reeve, R. Warren, G. Theron, and T. Niesler, “Automatic cough classification for tuberculosis screening in a real-world environment,” *Physiological Measurement*, vol. 42, no. 10, p. 105014, nov 2021.
- [3] G. P. Kafentzis, S. Tetsing, J. Brew, L. Jover, M. Galvosas, C. Chaccour, and P. M. Small, “Predicting tuberculosis from real-world cough audio recordings and metadata,” 2023.
- [4] M. Pahar, M. Klopper, R. Warren, and T. Niesler, “Covid-19 detection in cough, breath and speech using deep transfer learning and bottleneck features,” *Computers in Biology and Medicine*, vol. 141, p. 105153, 2022.
- [5] G. D. Yellapu, G. Rudraraju, N. R. Sripada, B. Mamidgi, C. Jalukuru, P. Firmal, V. Yechuri, S. Varanasi, V. S. Peddireddi, D. M. Bhimarasetty, S. Kanisetti, N. Joshi, P. Mohapatra, and K. Pamarthi, “Development and clinical validation of swaasa ai platform for screening and prioritization of pulmonary tb,” *Scientific Reports*, vol. 13, p. 4740, 2023.
- [6] B. T. Balamurali, H. I. Hee, S. Kapoor, O. H. Teoh, S. S. Teng, K. P. Lee, D. Herremans, and J. M. Chen, “Deep neural network-based respiratory pathology classification using cough sounds,” *Sensors (Basel)*, vol. 21, no. 16, p. 5555, Aug 2021.
- [7] C. Wall, L. Zhang, Y. Yu, and K. Mistry, “Deep recurrent neural networks with attention mechanisms for respiratory anomaly classification,” in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8.
- [8] A. Hidaka and T. Kurita, “Consecutive dimensionality reduction by canonical correlation analysis for visualization of convolutional neural networks,” in *Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications*, 2017, pp. 160–167.
- [9] World Health Organization, “Global tuberculosis report 2022,” 2022. [Online]. Available: <https://www.who.int/teams/global-tuberculosis-programme/tb-reports/global-tuberculosis-report-2022>

- [10] R. Bachu, S. Kopparthi, B. Adapa, and B. Barkana, *Voiced/Unvoiced Decision for Speech Signals Based on Zero-Crossing Rate and Energy*, 2010, pp. 279–282.
- [11] L. T. DeCarlo, “On the meaning and use of kurtosis,” *Psychological Methods*, vol. 2, no. 3, pp. 292–307, 1997.
- [12] A. Y. Ng, “Feature selection, l1 vs. l2 regularization, and rotational invariance,” in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. ICML ’04. New York, NY, USA: Association for Computing Machinery, 2004, p. 78.
- [13] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” *Computing Research Repository*, vol. abs/1511.08458, 2015.
- [14] T. R. Niesler, “Data engineering 414 course notes,” *Department of Electrical and Electronic Engineering, Stellenbosch University*, 2023.
- [15] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *Computing Research Repository*, vol. abs/1412.6980, 2014.
- [16] Hyfe Inc. (2023) Hyfe AI - detect and quantify cough. [Online]. Available: <https://www.hyfe.ai/>
- [17] C. for Disease Control and Prevention. (2016) Xpert mtb/rif factsheet. [Online]. Available: [https://www.cdc.gov/tb/publications/factsheets/testing/xpert\\_mtb-rif.htm](https://www.cdc.gov/tb/publications/factsheets/testing/xpert_mtb-rif.htm)
- [18] F. for Innovative New Diagnostics. (2006) Mgit manual. [Online]. Available: [https://www.finddx.org/wp-content/uploads/2023/02/20061101\\_rep\\_mgit\\_manual\\_FV\\_EN.pdf](https://www.finddx.org/wp-content/uploads/2023/02/20061101_rep_mgit_manual_FV_EN.pdf)
- [19] M. Bonnet *et al.*, “Bacterial culture through selective and non-selective conditions: the evolution of culture media in clinical microbiology,” *New Microbes and New Infections*, vol. 34, p. 100622, 2019.
- [20] K. Hajian-Tilaki, “Receiver operating characteristic (roc) curve analysis for medical diagnostic test evaluation,” *Caspian Journal of Internal Medicine*, 2013.
- [21] A. P. Bradley, “The use of the area under the roc curve in the evaluation of machine learning algorithms,” *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.
- [22] M. D. Ruopp, N. J. Perkins, B. W. Whitcomb, and E. F. Schisterman, “Youden index and optimal cut-point estimated from observations affected by a lower limit of detection,” *Biometrical Journal*, vol. 50, no. 3, pp. 419–430, 2008.

- [23] H. Liu, Q. Fu, L. Du, T. Zhang, G. Yu, S. Han, and D. Zhang, “Learning rate perturbation: A generic plugin of learning rate schedule towards flatter local minima,” in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, ser. CIKM '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 4234–4238.

# Appendix A

## Project Planning Schedule

Date	Proposed Work
13/06/2023	Skripsie kickoff meeting with Prof T.R. Niesler in which different possible topics will be discussed.
~ /06/2023	Decision made on skripsie topic. Research performed on topic field.
25/07/2023	Initial group meeting with the members of the DSP lab. Introduction and literature review report writing started.
14/08/2023	Submit GA document of ECSA outcomes signed by Prof T.R. Niesler Background and data chapter writing.
15/08/2023	Coding start with data pre-processing and initial feature extraction
25/08/2023	Logistic regression models coding to be implemented
08/09/2023	Train, validate and test logistic regression models
30/09/2023	Start with experimental setup in report CNN models coding to be implemented
07/10/2023	Train. validate and test CNN models
14/10/2023	Start with results and discussion of results chapters
22/10/2023	Start with finalising details of report
06/11/2023	Report submission deadline

# Appendix B

## Outcomes Compliance

**Table B.1:** ECSA outcomes compliance

Outcome	Description of outcome found in report
<b>GA 1:</b> Problem solving	This project aims to classify whether a patient has TB based on an audio recording of their cough. The procedure to achieve this is ill-defined, involves many intermediate steps, and is open-ended. Methods of signal processing, feature extraction, statistical classification, and empirical evaluation will have to be identified or designed, integrated, and tested. This is shown in Chapter 2: Literature Review, Chapter 3: Background, and Chapter 5: Experimental setup.
<b>GA 2:</b> Application of scientific and engineering knowledge	The feature extraction of the audio signal required a deep understanding of audio/signal processing to be applied correctly, as well as applying the knowledge of logistic regression, and a convolutional neural network was required to build the classifiers. This can be seen in Chapter 3: Background and in Chapter 5: Experimental setup where the knowledge was applied.
<b>GA 3:</b> Engineering design	A procedural design was followed from pre-processing of data, the performance of feature extraction, which is followed by building, evaluating, and testing different machine learning models. This can be seen in Chapter 4: Data, Chapter 5: Experimental setup, and Chapter 6: Experimental results.
<b>GA 4:</b> Investigations, experiments, and data analysis	Extensive experiments, investigation, and data analysis was. This can be seen in the hyperparameter optimizing process that required multiple experiments to be performed, data analysis on which models performed the best and the reasons for it. This can be seen in Chapter 5: Experimental setup and Chapter 6: Experimental results.
Continued on the next page	



Table B.1 – continued from previous page

Outcome	Description of outcome found in report
<b>GA 5:</b> Engineering methods, skills, and tools, including IT	This is a software-based engineering project, and the entire project was performed in the <b>Python</b> programming environment. The software was needed to perform data manipulation, create and train the different models as well as evaluate the performance. This is seen in Chapter 4: Data, 5: Experimental setup, and Chapter 6: Experimental results.
<b>GA 6:</b> Professional and technical communication	This final technical report is written in a technical and professional style with the correct use of language. This can be seen in the entire report.
<b>GA 8:</b> Individual work	All work was done by me and only supervision was given. This is seen throughout the entire report.
<b>GA 9:</b> Independent learning ability	This project required research on the field of audio/signal processing and the field of medical diagnosis using machine learning. The research on different machine learning models and how they are implemented was done. This is seen in Chapter 2: Literature review, Chapter 3: Background, and Chapter 4: Data.