الجمهورية الجزائرية الديموقراطية الشعبية

Ministry of Higher Education and Scientific Research

Yahia Fares University of Medea

Faculty of Sciences

Department of Mathematics and Computer Science



Graduation Project Report Submitted for Master of Computer science degree

Speciality: Engineering Systems and Web Technology

Respiratory Disorder Detection using Deep Learning

Proposed and directed by:

Dr. Ahmed Cherif Mazari

Prepared and presented by:

Ms. Amina Takdenti

2023-2024

Acknowledgement

First and foremost, I am deeply thankful and grateful to Allah for providing me with the patience and strength to persevere throughout this work.

I would like to extend my sincere gratitude to my supervisor, Ahmed Cherif Mazari, for his invaluable patience, encouragement, and guidance during the entire project.

Additionally, I would like to thank all the professors from whom I have learned over these past five years.

I acknowledge the use of ChatGPT for assistance in reformulating portions of this thesis.

Dedication

I dedicate this work to ALLAH that gave me the power and the strength to keep going until this day

To my first supporters in life my parents who were always here for me and by my side I love you mom and dad

To my sweetheart and my great sisters

To my friends

to everyone believed in me

and to me...

Abstract

This thesis investigates the use of deep learning techniques to detect respiratory disorders using audio recordings, aiming to improve early diagnosis. Respiratory disorders, such as Asthma, Chronic Obstructive Pulmonary Disease, pneumonia, Lung fibrosis, Pleural effusion, Bronchiectasis and heart failure, often go undiagnosed due to the lack of accessible diagnostic tools, leading to severe health complications. This study utilizes a dataset of 336 audio recordings categorized into eight classes. Techniques such as data segmentation, augmentation, and feature extraction using Mel Frequency Cepstral Coefficients and spectrograms are employed. Several deep learning models, including Convolutional neural network, long short-term memory, recurrent neural network, Time-delay neural network, Autoencoder, Attention and wavenet even combined deep learning models are applied and compared based on their performance metrics.

These deep learning models demonstrated high accuracy in classifying respiratory disorders, with Convolutional neural network, Time-delay neural network and wavenet models achieving above 99% in terms of accuracy, F1 score, recall, and precision.

Key words: Respiratory disorders, Deep learning, audio recordings, data segmentation, data augmentation, feature extraction, Mel Frequency Cepstral Coefficients, spectrograms.

ملخص

تستكشف هذه الدراسة استخدام تقنيات التعلم العميق للكشف عن الاضطرابات التنفسية باستخدام التسجيلات الصوتية، بهدف تحسين الكشف المبكر. الاضطرابات التنفسية، مثل الربو، والانسداد الرئوي المزمن، والالتهاب الرئوي، وتليف الرئة، وانصباب الجنب، وتوسع القصبات، وفشل القلب، غالباً ما تظل دون تشخيص بسبب نقص أدوات التشخيص المتاحة، مما يؤدي إلى مضاعفات صحية خطيرة. تستخدم هذه الدراسة مجموعة بيانات تتكون من 336 تسجيل صوتي مصنفة إلى ثماني فئات. تُستخدم تقنيات مثل تجزئة البيانات، والتكبير، واستخراج السمات باستخدام معاملات التردد القيقاني الميلي والطيفيات. يتم تطبيق ومقارنة عدة نماذج من التعلم العميق

أظهرت هذه النماذج دقة عالية في تصنيف الاضطرابات التنفسية تصل ل 99%

الكلمات المفتاحية: الاضطرابات التنفسية، التعلم العميق، التسجيلات الصوتية، التجزئة، التكبير، استخراج السمات

Table of Contents

LIST OF FIGURES **8**

LIST OF TABLES **10**

ACRONYMS **11**

GENERAL INTRODUCTION **12**

1 *CHAPTER 01:* MEDICAL SIDE **15**

1.1 INTRODUCTION 16
1.2 CHRONIC RESPIRATORY DISEASES: 16
1.2.1 Asthma *16*
1.2.2 Chronic obstructive pulmonary disease *18*
1.2.3 Bronchiectasis *19*
1.2.4 Lung fibrosis *20*
1.3 ACUTE RESPIRATORY DISEASE 21
1.3.1 Pneumonia *21*
1.3.2 Pleural effusion *22*
1.4 CONCLUSION 23

2 *CHAPTER 02:* GENERAL OVERVIEW **24**

2.1 INTRODUCTION 25
2.2 PART ONE 25
2.2.1 Data pre-processing *25*
2.2.2 Data splitting and balancing *29*
2.2.3 Model architecture *30*
2.2.4 Evaluation *36*
2.3 PART TWO 36
2.3.1 Machine learning: *39*
2.3.2 Deep learning: *40*
2.3.3 Combined: *40*
2.4 CONCLUSION: 41

3 CHAPTER 03: APPROACHES USED **42**

3.1 INTRODUCTION 43
3.2 THE DATASET 43
3.3 EXPERIMENTATIONS STEPS 45
3.3.1 Segmentation *45*
3.3.2 Data augmentation *46*
3.3.3 Feature extraction *48*
3.3.4 Data preparation *49*
3.4 MODELS 49
3.4.1 Deep learning models *50*
3.4.2 Combined models *56*
3.5 CONCLUSION 58

4 CHAPTER 04: RESULTS AND DISCUSSION **59**

4.1 INTRODUCTION: 60
4.2 ENVIRONMENT AND DEVELOPMENT TOOLS 60
4.2.1 Interactive computational environments *60*
4.2.2 Libraries *61*

4.3 RESULTS 62
4.3.1 Trained on row data *62*
4.3.2 Augmented data *67*
4.3.3 Segmented data *73*
4.4 DISCUSSION 83
4.5 CONCLUSION 84
5 GENERAL CONCLUSION **85**
6 REFERENCES **87**

List of figures

Figure 1:Respiratory system diagram 16
Figure 2: normal VS asthmatic lung 17
Figure 3: the difference between healthy bronchial tube and a COPD patient 18
Figure 4: the difference between healthy airway and a bronchiectasis patient 19
Figure 5: the difference between healthy lung and a lung fibrosis patient 20
Figure 6: the difference between healthy bronchial tube and a pneumonia patient 22
Figure 7: the difference between healthy lung and a pleural effusion patient 23
Figure 8: Audio Processing Workflow for Deep Learning in Respiratory Disorder Detection 25
Figure 9: the difference between segmented audio and an original one 26
Figure 10: MFCC steps 28
Figure 11: Spectrogram steps 29
Figure 12: Example of CNN architecture in audio classification[6] 31
Figure 13: LSTM architecture 32
Figure 14: Types of RNNs 32
Figure 15: TDNN architecture[11] 33
Figure 16: A simple Attention mechanism 34
Figure 17: Architecture of general autoencoder[15] 35
Figure 18: workflow for processing audio data to detect respiratory disorders using deep learning models 43
Figure 19:number of classes in the dataset 44
Figure 20: visualisation of each label 44
Figure 21: audio after implementing noise reduction and segmentation 46
Figure 22: number of classes before and after augmentation 48
Figure 23: CNN Performance Evaluation on row data 63
Figure 24: LSTM Performance Evaluation on row data 63
Figure 25: RNN Performance Evaluation 64
Figure 26: TDNN Performance Evaluation on row data 65
Figure 27: attention Performance Evaluation on row data 65
Figure 28: Autoencoder Performance Evaluation on row data 66
Figure 29: Wavenet Performance Evaluation on row data 67
Figure 30: CNN Performance Evaluation on augmented data 68
Figure 31: LSTM Performance Evaluation on augmented data 69
Figure 32: RNN Performance Evaluation on augmented data 70
Figure 33: TDNN Performance Evaluation on augmented data 70
Figure 34: Attention Performance Evaluation on augmented data 71
Figure 35: Autoencoder Performance Evaluation on augmented data 72
Figure 36: Wavenet Performance Evaluation on augmented data 72
Figure 37:CNN Performance Evaluation on segmented data 74
Figure 38: LSTM Performance Evaluation on segmented data 74
Figure 39: RNN Performance Evaluation on segmented data 75
Figure 40: TDNN Performance Evaluation on segmented data 76
Figure 41: attention Performance Evaluation on segmented data 76
Figure 42:Autoencoder Performance Evaluation on segmented data 77
Figure 43: Wavenet Performance Evaluation on segmented data 78
Figure 44: CNN+RNN Performance Evaluation on segmented data 79

Figure 45: CNN+ Attention Performance Evaluation on segmented data 79
Figure 46: CNN +Wavenet Performance Evaluation on segmented data 80
Figure 47: CNN+ RNN +Attention Performance Evaluation on segmented data 81
Figure 48: Evaluation on row, augmented and segmented data 82

List of tables

Table 1: Most relevant works 39
Table 3: Summery of the CNN model 50
Table 4:Summery of LSTM model 51
Table 5: summery of the RNN model 52
Table 6: TDNN model 53
Table 7: summery of attention model 54
Table 8: Autoencoder summery 54
Table 9: Wavenet summery 55
Table 10: Evaluation on row data 67
Table 11: Evaluation on augmented data 73
Table 12: Evaluation on segmented data 81
Table 13: Comparison between several deep learning models trained on different data 83

Acronyms

AI Artificial Intelligence

DL Deep Learning

ML Machine Learning

COPD Chronic Obstructive Pulmonary Disease

MFCC Mel Frequency Cepstral Coefficients

ANN Artificial Neural Network

CNN Convolutional Neural Network

LSTM Long Sort-Term Memory

RNN Recurrent Neural Network

TDNN Time Delay Neural Network

CRNN Convolutional Neural Network

SVM Support Vector Machine

KNN k-nearest neighbours

DT Decision Tree

FTT Fast Fourier Transform

DCT Discrete Cosine Transform

GPU Graphics Processing Unit

N Healthy individual

ReLu Rectified Linear Unit

API Application Programming Interface

STFT Short-Time Fourier Transform

General Introduction

General Introduction

Breath sounds provide significant insights into a patient's health, revealing vital details with each inhale and exhale about their respiratory condition. Respiratory disorders which affect individuals of all ages including infants, disrupt normal breathing and are broadly classified into two categories: acute and chronic. The World Health Organization (WHO) identified respiratory diseases as the third leading cause of death globally in 2019. These conditions are especially prevalent in lower-income countries, where environmental factors such as air pollution, exposure to fumes and chemicals commonly trigger them. Moreover, these regions often suffer from limited access to medical care, exacerbating the impact of respiratory diseases and leading to increased rates of mortality.

The respiratory system itself is divided into two functional zones: the conducting zone which extends from the nose to the bronchioles and serves as a pathway for inhaled gases, and the respiratory zone from the alveolar duct to the alveoli where gas exchange occurs. Acute respiratory disorders such as pneumonia and pleural effusion manifest quickly and require immediate care while chronic conditions like asthma, chronic obstructive pulmonary disease (COPD), bronchiectasis, and lung fibrosis develop slowly and require long-term management. This distinction underscores the diverse nature of respiratory diseases and the critical importance of understanding and addressing them in healthcare.

As technology rapidly advances, particularly with the development of artificial intelligence (AI), its applications in various fields are proving to be highly beneficial to human beings. One of the most critical areas where AI is making a significant impact is healthcare, specifically in diagnosing various diseases. AI is increasingly utilized in analysing radiographic images, audio recordings, and other medical data. Among these applications, the use of AI to diagnose respiratory diseases through the analysis of breath sounds is particularly noteworthy. This innovative approach leverages the subtle nuances in breath sounds, offering a non-invasive and efficient diagnostic tool.

The purpose of this study is to diagnose various respiratory disorders including asthma, COPD, pneumonia, heart failure, bronchiectasis, pleural effusion, and lung fibrosis during an attack. Doctors often struggle to diagnose these conditions when the patient is not exhibiting symptoms. This study aims for the early detection of these disorders, which can significantly impact patient outcomes. To achieve better performance, employing several deep learning techniques.

This report is organized as follow:

- Chapter 01: **Medical side**
  In this chapter, A general review is provided on each disease, including its symptoms, causes, and treatments.

- Chapter 02: **General overview**
  In this chapter, the most relevant studies related to the research are presented, detailing the models used, the datasets employed, and the results achieved. Additionally, a general overview of the deep learning approaches proposed for classifying respiratory disorders is provided. This includes a detailed description of all the steps followed in pre-processing, building the proposed models, and their evaluation.
- Chapter 03: **Proposed approaches**
  The chapter explores the various approaches used, providing detailed descriptions of the parameters involved. Additionally, it explores the methodologies and techniques implemented in the research, highlighting their effectiveness and the rationale behind selecting specific models and pre-processing steps.
- Chapter 04: **Results and discussion**
  This chapter provides the results of the performance evaluation of deep learning approaches, including discussions and comparisons of individual models as well as combined models.

Chapter 01: **Medical side**

1.1 Introduction

Respiratory disorders are diseases that affect the ability of breathing in the respiratory system which is divided into two functional zones. conducting zones (nose to bronchioles) form a path for conduction of the inhaled gases and respiratory zone (alveolar duct to alveoli) where the gas exchange takes place. These disorders can be either acute, such as pneumonia and pleural effusion as well as chronic respiratory diseases such as asthma, chronic obstructive pulmonary disease, bronchiectasis and lung fibrosis.

These disorders are usually diagnosed by Pulmonary Function Tests (PFTs), but sometimes the diagnosis is not 100% accurate outside of an attack.

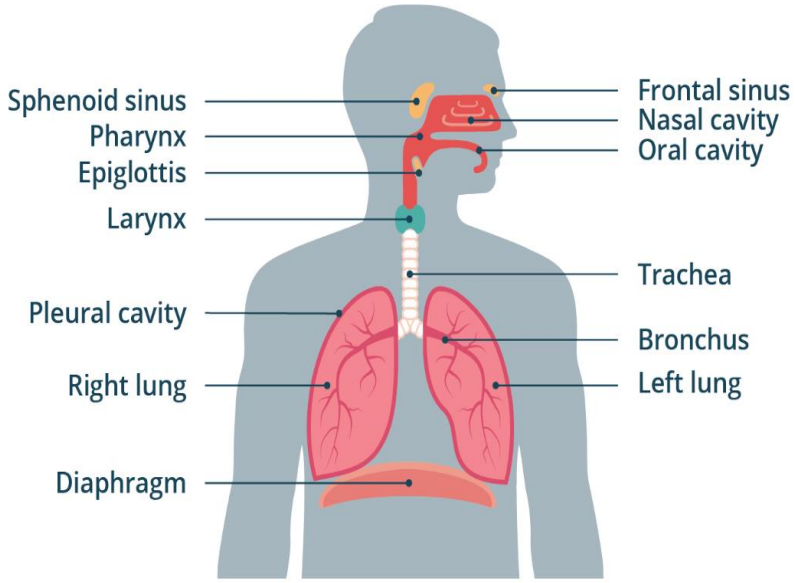


Figure 1:Respiratory system diagram[1]

1.2 Chronic respiratory diseases

1.2.1 Asthma

According to the World Health Organization (WHO), Asthma is a chronic respiratory disease that affect people of all ages. It is caused by inflammation and muscle tightening around the airways which makes it harder to breathe, Symptoms are coughing, wheezing, shortness of breath and chest tightness. These symptoms can have different severity and can come and go over time. Currently, nearly 339 million people live with asthma. In 2016, the WHO estimated that asthma caused 417,918 deaths globally.

[1] https://www.healthdirect.gov.au/respiratory-system (June 2024)

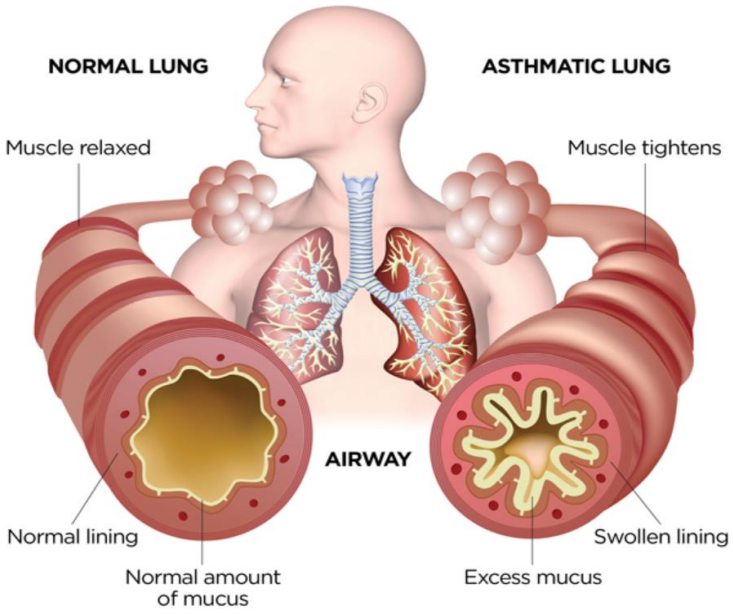


Figure 2: normal VS asthmatic lung[2]

- Symptoms
  There is individual variety Symptoms are often nocturnal or during exercise, the common symptoms of asthma include:
  - a persistent nocturnal cough
  - wheezing when exhaling and sometimes when inhaling
  - shortness of breath or difficulty breathing sometimes even when resting
  - chest tightness making it difficult to breathe deeply.
- Causes
  - Asthma is more likely hereditary.
  - Asthma is more likely in allergic people such as eczema and rhinitis (hay fever).
  - Urbanization is associated with increased asthma prevalence, probably due to multiple lifestyle factors.
  - Events in early life affect the developing lungs and can increase the risk of asthma. These include low birth weight, prematurity, exposure to tobacco smoke and other sources of air pollution, as well as viral respiratory infections.
  - Exposure to a range of environmental allergens and irritants are also thought to increase the risk of asthma, including indoor and outdoor air pollution, house dust mites, moulds and occupational exposure to chemicals, fumes or dust.
  - Overweight or obesity are at a greater risk of asthma.
- Treatment
  Asthma is a chronic condition. During asthma attacks, patients must use an inhaler to manage to breath. Preventive measures such as avoiding dust, pollution, smoke, tobacco, and stress can help reduce the frequency and severity of asthma attacks. Identifying and avoiding individual triggers plays a crucial role in maintaining better control over the condition.

[2] https://community.aafa.org/blog/what-happens-in-your-airways-when-you-have-asthma (June 2024)

1.2.2 Chronic obstructive pulmonary disease

Chronic Obstructive Pulmonary Disease (COPD) is a common lung disease that restricts airflow and leads to breathing difficulties. Often referred to as emphysema or chronic bronchitis, COPD cause lung damage or excessive phlegm build-up. In 2019, it was the third leading cause of death globally, responsible for 3.23 million deaths. Nearly 90% of COPD-related deaths among individuals under 70 occur in low- and middle-income countries.

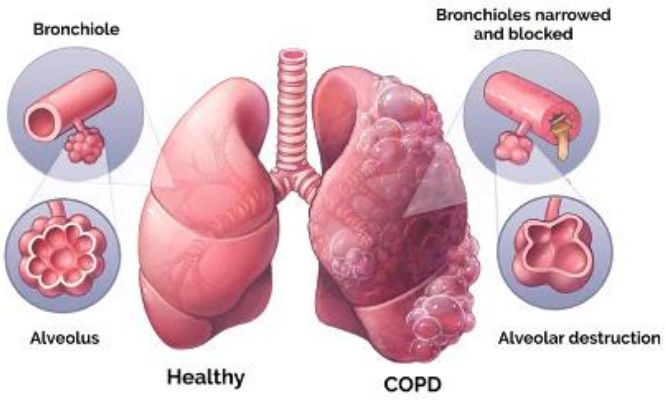


Figure 3: the difference between healthy bronchial tube and a COPD patient[3]

- Symptoms
  The most common symptoms of COPD are difficulty breathing, chronic cough (sometimes with phlegm) and fatigue.
  People with COPD also have a higher risk for other health problems. such as:
  - lung infections like the flu or pneumonia
  - lung cancer
  - heart problems
  - weak muscles and brittle bones
  - depression and anxiety.
- Causes
  - tobacco exposure from active smoking or passive exposure to second-hand smoke.
  - occupational exposure to dusts, fumes or chemicals.
  - indoor air pollution: biomass fuel (wood, animal dung, crop residue) or coal is frequently used for cooking and heating in low- and middle-income countries with high levels of smoke exposure.
  - early life events such as poor growth in utero, prematurity and frequent or severe respiratory infections in childhood that prevent maximum lung growth.
  - asthma in childhood.
  - a rare genetic condition called alpha-1 antitrypsin deficiency, which can cause COPD at a young age.

[3] https://www.mrinz.ac.nz/programmes-2/copd (June 2024)

- Treatment
  Chronic obstructive pulmonary disease (COPD) is chronic but it can improve with lifestyle changes like quitting smoking, avoiding pollution, and receiving vaccinations. Treatments include medications, oxygen therapy, and pulmonary rehabilitation. Several options are available the primary treatment with inhaled medications that open up airways and reduce inflammation. These approaches help improve breathing and enhance the quality of life for individuals living with COPD.

1.2.3 Bronchiectasis

Bronchiectasis is an abnormal widening of one or more airways. Normally small glands in the airway linings produce a modest amount of mucus, which is moisturizer and traps dust and other particles from inhaled air. However, in bronchiectasis, the widened airways lead to increased mucus production which can accumulate and pool in these expanded areas. This excess mucus makes the airways more susceptible to infections.

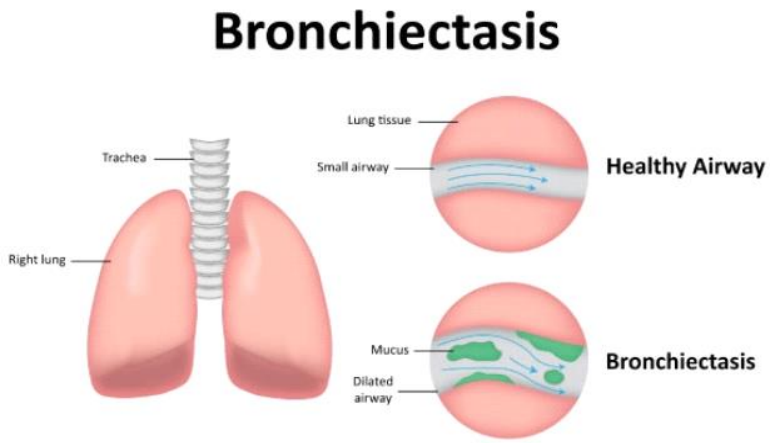


Figure 4: the difference between healthy airway and a bronchiectasis patient[4]

- Symptoms
  The main symptom of bronchiectasis is a cough that produces a lot of sputum, wheeziness, tiredness and poor concentration.
- Causes
  Bronchiectasis is a lung disease primarily caused by chronic infections that lead to inflammation in the airways. It is characterized by limited airflow, often due to growths or noncancerous tumours obstructing the airways. In some cases, bronchiectasis can develop during childhood following a severe lung infection.
- Treatment
  Bronchiectasis has no cure since it is a chronic condition but effective management is crucial to control its symptoms. The primary goals of treatment are to reduce respiratory infections and manage bronchial mucus production. Treatment options for bronchiectasis include:

[4] https://www.medicoverhospitals.in/diseases/bronchiectasis/ (June 2024)

- Chest physiotherapy to help clear mucus from the lungs.
- Antibiotics to treat and prevent infections.
- Techniques to drain secretions from the airways.
- Addressing any underlying conditions that might be contributing to the bronchiectasis.
- Ensuring adequate hydration, which helps thin the mucus, making it easier to clear.

These strategies are essential to improve life quality and to prevent complications associated with the disease.

1.2.4 Lung fibrosis

Pulmonary or lung fibrosis is a condition characterized by an excessive build-up of fibrotic tissue in the lungs, which can stem from various causes, including environmental factors and autoimmune diseases. This thickening of lung tissue impairs respiratory function, potentially leading to severe complications such as respiratory failure and death.

Early diagnosis and prompt treatment are critical to managing this condition effectively. Treatment options typically focus on reducing inflammation, slowing the progression of tissue damage and maintaining adequate lung function. In more severe cases, lung transplantation may be considered. The main goals are to improve breathing capacity and enhance the quality of life for affected individuals.

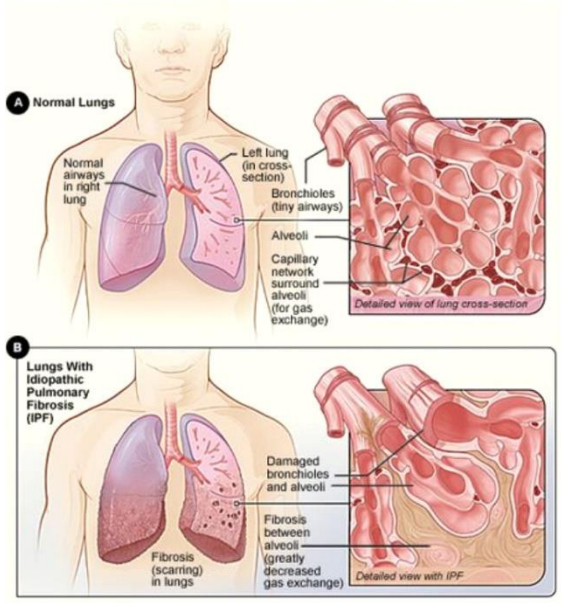


Figure 5: the difference between healthy lung and a lung fibrosis patient[5]

- Symptoms

  Individuals with long term Idiopathic Pulmonary Fibrosis (IPF) may not show any symptoms but as scarring continues to progress within the lungs, patient might have:

  - Difficulty in breathing (Dyspnea)

[5] https://www.physio-pedia.com/Pulmonary_Fibrosis (June 2024)

- An unproductive cough that is persistent
- Shortness of breath (especially when engaging in activities such as walking)
- Greater levels of fatigue
- Muscular pain (myalgias)
- Joint pain (arthralgias)
- Unexplained weight loss
- Clubbing in fingers and toes

- Treatment

  Managing chronic pulmonary diseases typically includes:
  - Lifestyle Changes: Encouraging smoking cessation and ensuring vaccinations against influenza and pneumococcal diseases.
  - Pulmonary Rehabilitation: Exercises and education to improve lung function.
  - Avoiding Harmful Agents: Such as certain medications that can worsen fibrosis.
  - Oxygen Therapy: For patients with severe hypoxemia to help maintain oxygen levels.
  - Medications: Corticosteroids may be used for specific conditions like sarcoidosis.
  - Lung Transplant: Considered for severe cases, with early evaluation for transplant eligibility.

  Treatment strategies are personalized, focusing on slowing progression and improving quality of life.

1.3 Acute respiratory disease

They are conditions that develop rapidly and typically have a short duration. These diseases often present with severe symptoms that require immediate medical assistance. Common acute respiratory diseases include:

1.3.1 Pneumonia

Pneumonia is a severe respiratory infection mainly caused by viruses or bacteria. It affects people of all ages with different degrees of severity. It's particularly dangerous for children and is the leading cause of death from infection in children under five worldwide. In 2017, pneumonia caused the deaths of more than 808,000 children under five, representing 15% of all deaths in this age group.

Adults over 65 and individuals with existing health problems are also at higher risk for pneumonia. The disease affects the lungs, specifically the small sacs known as alveoli that fill with air when you breathe. In cases of pneumonia, these alveoli fill with pus and fluid, making breathing painful and reducing oxygen intake. Pneumonia is contagious with direct contact with infected people.

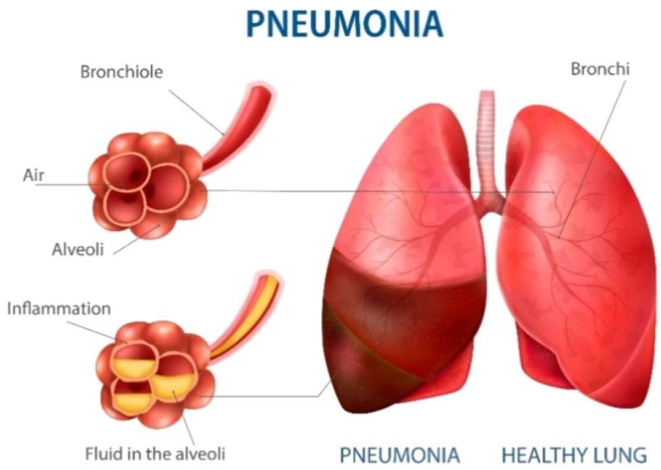


Figure 6: the difference between healthy bronchial tube and a pneumonia patient[6]

- Symptoms
  Depending on the severity of the pneumonia, signs and symptoms may include:
  - Cough
  - Shortness of breath
  - Fever, sweating and shaking chills
  - Fatigue
  - Chest pain
  - Nausea, vomiting or diarrhea
  - Confusion, especially in older adults.
- Treatment
  Pneumonia is curable and can be treated with antibiotics if it is caused by bacteria. Additionally, getting plenty of rest and staying hydrated can help speed up recovery. To reduce the risk of pneumonia, vaccinations are essential. Addressing environmental factors, practicing regular handwashing, and maintaining a healthy lifestyle are also crucial preventive measures.

1.3.2 Pleural effusion

A pleural effusion is an excessive build-up of fluid in the pleural space (area between the layers of tissue lining the lungs and the chest cavity). This condition suggests an imbalance between the production and removal of pleural fluid. Although a pleural effusion itself is not a specific disease, it is a symptom that indicates the presence of underlying health problem. It can occur with various lung and pleural disorders, as well as other systemic conditions. Essentially, pleural effusions can be a symptom of different medical problems, highlighting the need to identify and treat the root cause[1].

[6] https://continentalhospitals.com/blog/common-symptoms-of-pneumonia-how-can-i-prevent-it/ (June 2024)

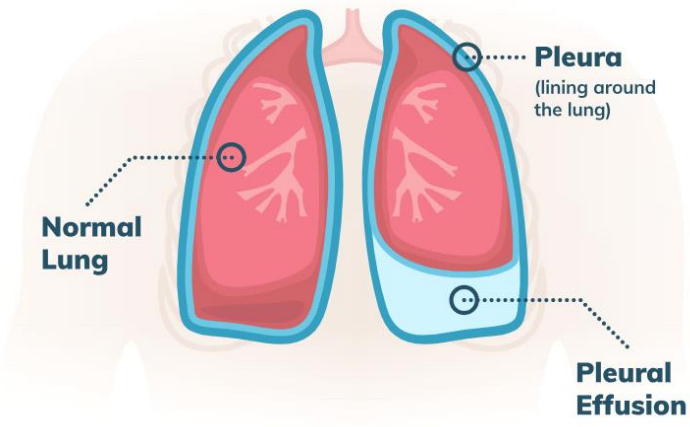


Figure 7: the difference between healthy lung and a pleural effusion patient[7]

- Symptoms
  Pleural effusion symptoms include:
  - shortness of breath
  - chest pain
  - coughing and fatigue

  Some patients may not have symptoms. In these cases, the effusion may be found incidentally.
- Causes
  Heart failure, cancer, and pneumonia are the most common conditions that lead to pleural effusions. These illnesses can cause an effusion by either increasing the production of pleural fluid, reducing its drainage, or both. Additionally, a variety of other medical conditions can also result in pleural effusions.
- Treatment
  Treatments for pleural effusion focus on treating the underlying cause. For example, doctors use diuretics to treat effusions caused by heart failure and antibiotics for those caused by infections. For malignant effusions, such as those from cancer, the treatment often involves draining the fluid.

1.4 Conclusion

As a conclusion, the respiratory diseases mentioned in this chapter represent some of the most prevalent and deadly health challenges globally, causing millions of deaths annually. These conditions, disproportionately affect poorer regions where access to diagnostic services and effective treatments is often limited. The lack of healthcare infrastructure and resources in these areas intensifies the impact of such diseases, leading to higher mortality rates and significant public health issues.

[7] https://www.mesothelioma.com/mesothelioma/symptoms/pleural-effusion/ (June 2024)

Chapter 02: **General overview**

2.1 Introduction

In recent decades, artificial intelligence has revolutionized many fields with healthcare standing out as a prime area of transformation. AI technologies, particularly deep learning, have become integral in diagnosing and treating diseases offering new methodologies that were unimaginable just a few years ago. This chapter explores

the application of deep learning in medical diagnostics, focusing on respiratory disorder detection through audio analysis a challenging but increasingly vital area in medical AI, as well as a review of more relevant work in this field.

2.2 Part one

Respiratory sounds hold critical clues about patient health but using this data often involves overcoming significant obstacles, primarily due to the lack of data, in order to effectively tackle these challenges, several key techniques are used such as segmentation, data augmentation and feature extraction.

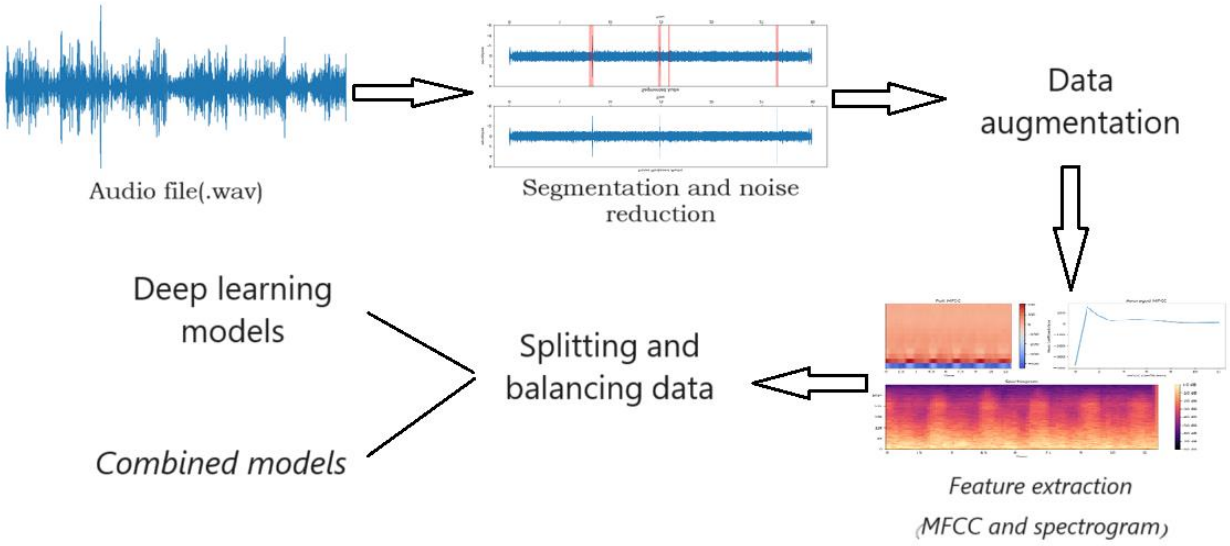


Figure 8: Audio Processing Workflow for Deep Learning in Respiratory Disorder Detection

2.2.1 Data pre-processing

2.2.1.1 Segmentation

it is a critical step in the pre-processing of audio data, the aim of audio segmentation is to divide the audio signal into small segments or frames so that the entities may be easily identified[2].

One of the segmentation techniques is Energy-based segmentation that divide the audio based on the energy level in different parts of it such as the beginning and the end of the breathing sound, then calculates the energy for each frame using:

$$E = \sum(audio[i: i + frame_length]^2)1$$

Where $audio[i: i + frame_length]$ represents a segment of the audio data starting at index i and extending up to $i + frame_length$.

All the energy values are normalized to a scale from 0 to 1 by normalization formula:

$$normalized_energy = \frac{energy - min(energy)}{max(energy) - min(energy)} \quad 2$$

Then using a Thresholding that determines whether a specific frame in the audio qualifies as an active segment containing significant sounds such as breath sounds or should be classified as silence or background noise, for example Frames with energy above this threshold mark the start of a new segment if the previous frame was below the threshold, suggesting a transition from low energy (silence) to higher energy (sound).This method ensures that only the relevant audio segments are retained for further analysis.
The figure below shows the audio before and after the segmentation

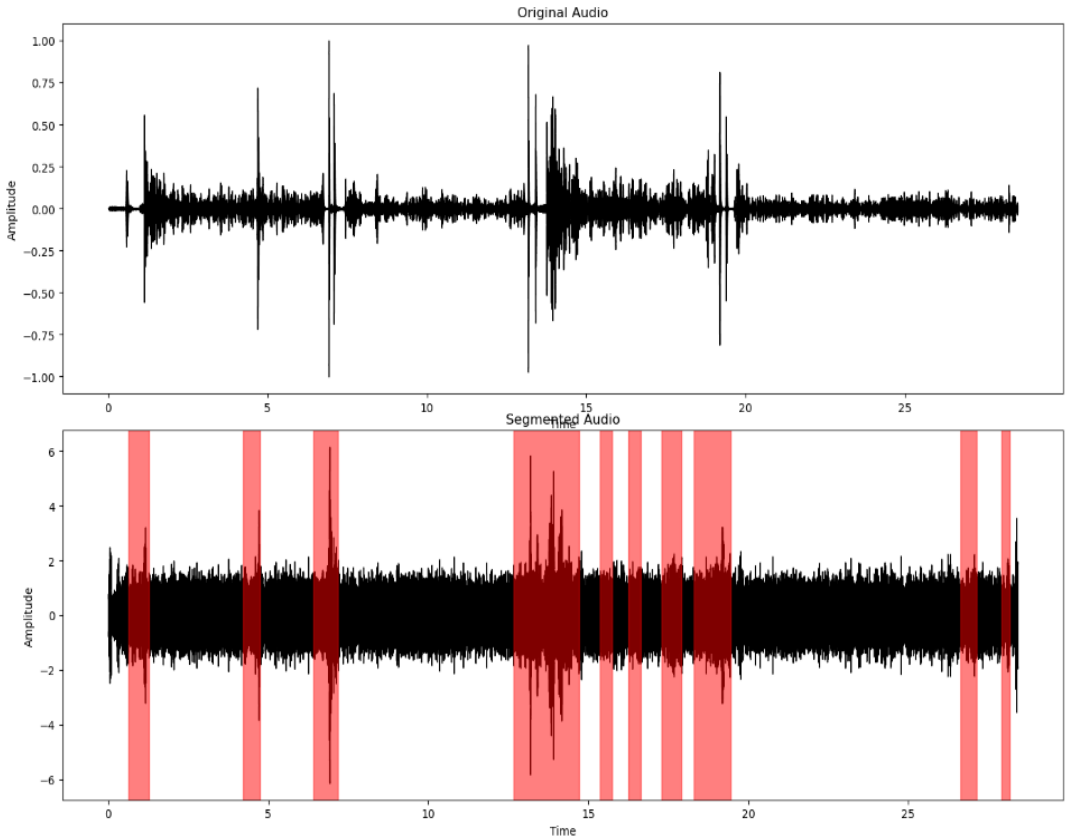


Figure 9: the difference between segmented audio and an original one

2.2.1.2 Data augmentation

It is a technique used to artificially increase the size of a dataset by creating modified versions of data points which helps address the problem of insufficient data and class imbalance in training datasets. This method is useful in areas such as speech, audio and imaging while It's important to apply augmentation techniques carefully to make sure that the quality of the data is not lost especially when working with such specific data such as breath sounds. Some of the common techniques used in audio data augmentation are:

- Voice adjustment
  This technique modifies the loudness of an audio recording by adjusting the volume using volume factors where less than 1 decrease the volume while greater than 1 increase it. By training models with audio data at varied volume levels the model becomes more adept at handling real-world scenarios where the loudness of speech or other sounds can vary significantly.
- Noise
  Incorporating different types of background noise train models to be more effective in the real world where the higher value means louder voice.
- Time shifting
  this technique works on moving the audio signals forward or backword, the forward one is safer because adding a period of silence at the beginning of the audio will not affect the quality of the audio.
- Dynamic range compression
  It is well known that audio recordings can have uneven levels; for example, respiratory sounds may vary significantly as breathing intensity changes. Dynamic range compression addresses this by reducing the volume of loud sounds or amplifying quieter ones, thus compressing the audio signal's dynamic range.
- Mild pitch shifting
  This technique involves altering the pitch or frequency of an audio signal without changing its duration by either speeding up or slowing down the playback and then adjusting the length to match the original, the pitch can be raised with positive values or lowered with negative ones.
- Equalization
  It involves adjusting the balance of frequencies within an audio signal to enhance sound quality or achieve a desired effect by increasing or decreasing the amplitude of specific frequency bands, equalizers can help clarify speech, reduce noise, or make certain sounds more prominent. This technique is especially useful in managing audio signals where some frequencies are overpowering or underrepresented allowing for a more balanced and pleasing auditory experience.

2.2.1.3 Feature extraction

There are numerous techniques for feature extraction in audio processing, but the most commonly used are spectrograms and Mel Frequency Cepstral Coefficients.

- Mel Frequency Cepstral Coefficients
  It is a technique that used in speech recognition based on frequency domain using the Mel scale which is based on the human ear scale[3]. [4] The MFCC is calculated by 5 steps which are:
  - Pre-Emphasis: The first step involves applying a hight-pass filter to the audio signal to increase the hight-frequency components that are generally suppressed in the recording process, thereby modifying the energy distribution across frequencies.
  - Signal framing and windowing: Dividing the audio signal into short frames around 20-30 milliseconds long with a 10-milliseconds overlap with the next frame.

- Power spectrum: for each frame the power spectrum is calculated using the Discrete Fourier Transform (DFT)

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{-\frac{2\pi jnk}{N}}, k = 1,2, \dots N-1 \quad (3)$$

Where $x(n)$is discrete signal and N is the length of the signal.

- Mel filter bank: This involves applying a bank of triangular band-pass filters based on the Mel scale that capture frequencies relevant to human hearing, providing a nonlinear representation of the audio signal.
- Discrete Cosine Transform (DCT): After applying the Mel filter bank, the log-transformed filter outputs are processed using the DCT to de-correlate the filter bank coefficients and emphasize the most significant components for analysis

$$X(k) = \sum_{n=0}^{N-1} x_n * cos(2\pi jnk/N), k = 1,2, \dots N-1 \quad (4)$$

where x_n is discrete signal and N is the length of the signal.
The figure below shows the steps of the MFCC

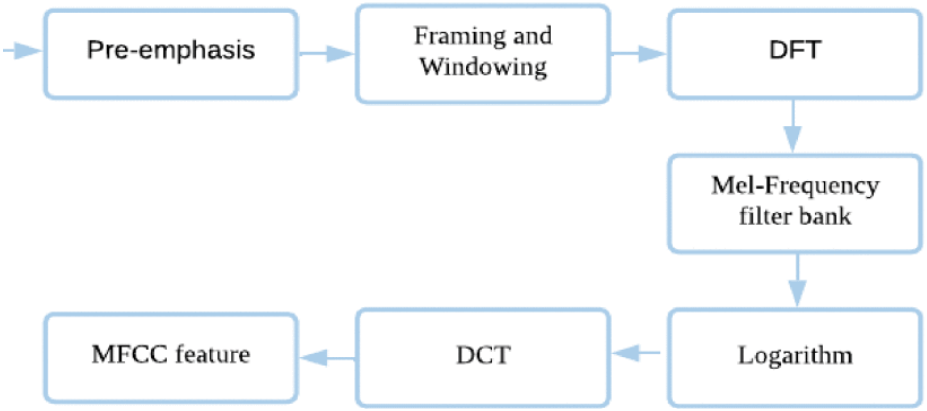


Figure 10: MFCC steps

- Spectrograms
It is a visual representation of the spectrum of frequencies in a sound or other signal as they vary with time, this technique is useful in audio analysis and speech processing.
  - Signal division: Starting with segmenting the audio signal into short frames or segments same as the pre-emphasis in MFCC.
  - Windowing: the abrupt starts and ends cause spectral leakage so in order to minimize it, each segment is multiplied by a window function generally Hamming or Hanning, these functions taper the beginning and end of each frame to reduce artifacts in the Fourier transform.

Hamming window: $w(n) = 0.54 - 0.46cos\left(\frac{2\pi n}{N-1}\right)$ (5)

Where $w(n)$ is the window function, *n* is sample index and *N* is the total number of samples in the window.

Hanning window: $w(n) = 0.5(1 - cos\left(\frac{2\pi n}{N-1}\right))$ (6)

Where $w(n)$ is the window function, n is sample index and N is the total number of samples in the window.

- Fourier transform: for each windowed frame a Fast Fourier Transform (FFT) is applied in order to translate it from the time domain to frequency domain. where the FFT is calculated by:

$$X(k) = \Sigma_{n=0}^{N-1} x(n) e^{-jnk\frac{2\pi}{N}} \quad (7)$$

where: $X(k)$ is the Fourier Transform of $x(n)$
$x(n)$is the signal in the time domain
N is the total number of samples in the frame
n is the current sample index
k is the index of frequency component (0 to N-1)
j is the imaginary unit

- Spectrogram generation: The results of magnitudes are plotted on a graph where one axis represents time and the other frequency. this visualization helps in analysing the behaviour of different frequencies over time providing visions into the audio content's characteristics.
The steps to compute Spectrograms are shown in the figure below



Figure 11: Spectrogram steps

- Combined
Combining features from different extraction methods can greatly improve performance in audio processing tasks. the mixture of Mel Frequency Cepstral Coefficients and spectrogram features is common because of their complementary nature where they cooperatively capture both the perceptual and dynamic aspects of audio signals, where MFCCs focus on the spectral envelope which is stable and distinct among different sounds making it essential for capturing the textural characteristics of it, on the other hand Spectrograms provide detailed insights into transient features and fine-grained spectral varies over time providing a dynamic representation of the sound's development.
The features extracted from both MFCC and spectrogram are concatenated into a single feature vector for each audio frame creating a rich dataset that can be utilized in various machine learning and deep learning models.

2.2.2 Data splitting and balancing

2.2.2.1 Imbalance problem

Handling an imbalanced dataset is a significant challenge in classification tasks, often leading to poor predictive performance on the minority classes. There are two primary techniques to address this issue:

- Oversampling the Minority Class: Increasing the number of samples in the minority class by duplication.
- Under-sampling the Majority Class: Reducing the number of samples in the majority class.

The choice of the right technique depends on the specific domain and the nature of the data. In fields such as respiratory health where missing a rare condition could be detrimental, oversampling the minority class is often preferred. This ensures that the model is exposed to sufficient examples of less common conditions, thereby improving its diagnostic accuracy.

2.2.2.2 Splitting data

To divide the data into distinct sets to train and evaluate the model effectively.

- Training Set: It is used to train the model, usually the largest portion (e.g., 70% or 80% of the data).
- Validation Set: It is used to fine-tune model parameters and prevent overfitting. This set can be used to select the best model during training (e.g., 10% or 15% of the data).
- Test Set: It is used to assess the performance of the model after the training process, simulating how it would perform under real-world conditions (e.g., 10% or 15% of the data).

Now that the data has been sufficiently prepared and balanced, it is ready to be used for training various deep learning models.

2.2.3 Model architecture

2.2.3.1 Convolutional Neural Network

Convolutional neural network (CNN)[5] is the most used model in deep artificial neural networks that can be used in various fields such as imaging, video recognition, speech, and natural language processing. These networks have been inspired by biological processes. A CNN contains various components, [6] which are:

a. Layers:

- Convolutional layer: This layer applies a set of learnable filters, and the output is fed to other convolutional layers. The number of convolutional layers depends on network requirements.
- Pooling layer: This layer reduces the number of connections between convolutional layers. The most common types are max and average pooling.
- Fully connected layers: These layers are present in varying numbers in a CNN, generally positioned right before the output layers. Every neuron in this layer is connected to every neuron in the previous layer. It is used to linearly transform the network's output but can be replaced or converted into convolutional layers to turn the system into a fully convolutional network.
- The softmax layer: It is generally a linear layer with a softmax classifier that converts the activations into values between 0 and 1. It helps determine the final outputs of the CNN, such as classifications or recognitions. It is represented by:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{k} e^{z_k}} \quad 8$$

Where z is a vector of input units, j is an index for output values while k is an index to input values with the total number of elements in j being K.

b. Activation Function: The aim of activation functions is to introduce non-linearity, ensuring that the activation of one neuron can influence multiple others. The most common one is ReLu, which retains the positive values of the input and turns the negative ones to 0. It is represented as:

$$A[i,j,k] = max(Z[i,j,k], 0)$$ 9

where: i, j and k are indices that refer to the specific elements in the tensors being processed by the ReLu function, A is the output tensor and Z is the input tensor that contains the data from the previous layer before activation.

c. Regularization function: The use of many parameters in CNNs leads to overfitting, so regularization techniques are employed. There are many techniques, but the most common ones include Dropout where randomly selected neurons are ignored during training, reducing the chance of overfitting.

The model architecture is represented in the figure below.

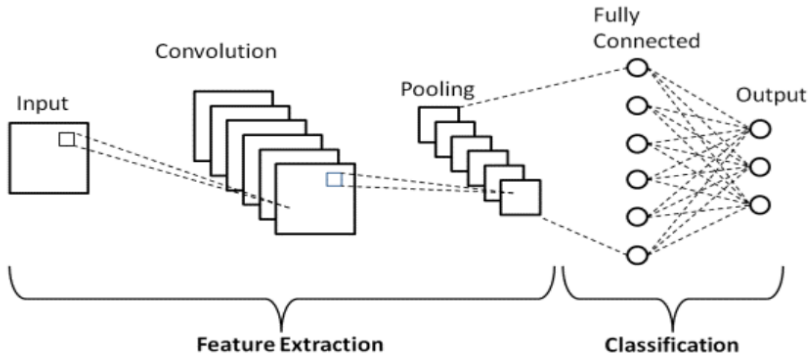


Figure 12: Example of CNN architecture in audio classification[7]

2.2.3.2 Long Short-Term Memory

Long Short-Term Memory (LSTM)[8] model consists of a set of recurrently connected subnets known as memory blocks. Each block contains one or more self-connected memory cells and three multiplicative units[9]. LSTM cells are designed to keep information over long periods effectively. They not only repeat the previous output but also keep an internal cell state, acting as a vector for long-term memory. This dual memory access to both short-term (hidden state) and long-term (cell state) memory, justifies their name: long short-term memory cells. The LSTM uses the three gates [10]:

- Forget gate: Regulates forgetting of previous information. If a scene ends, we want to forget all the information associated with the scene.
- Input gate: Controls the relevance and saving of new information. When a new scene is introduced, the input gate writes to memory the critical information to remember about the scene.
- Output gate: Joins the long-term information with the short-term information to give a better prediction about the next hidden unit/output.

The model architecture is represented in the figure below.

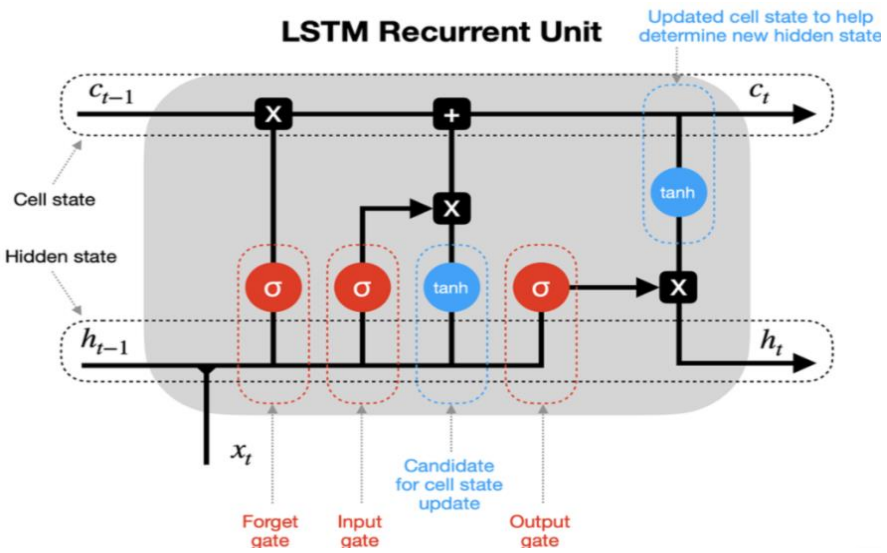


Figure 13: LSTM architecture[8]

2.2.3.3 Recurrent Neural Network (RNN)

Recurrent Neural Network(RNN)[11] is a type of neural network characterized by loops in its connections allowing it to keep information. This looping structure enables the RNN to process sequences of data making it particularly effective for tasks like speech and text processing. RNNs use feedback connections to send activations back through the network which helps in learning and remembering information across sequences. These networks are adept at handling sequential tasks such as speech recognition, language translation, and even music processing, as well as applying their capabilities to non-sequential tasks like image and video captioning[12]. The model architecture is represented in the figure below.

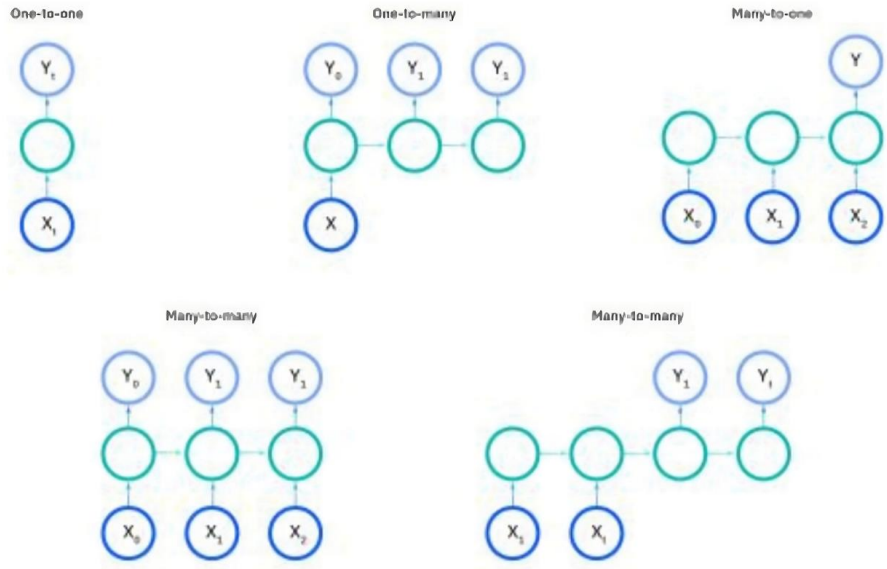


Figure 14: Types of RNNs[9]

[8] LSTM Recurrent Neural Networks — How to Teach a Network to Remember the Past | by Saul Dobilas | Towards Data Science (June 2024)

2.2.3.4 Time Delay Neural Network (TDNN)

Time-Delay Neural network (TDNN)[13] utilizes a hierarchical architecture to model temporal contexts effectively. Each layer in a TDNN operates at a distinct temporal resolution, enhancing the network's ability to process sequential data over time. The outputs from one layer's activations are spliced and fed as inputs into the subsequent layer, allowing each layer to operate within an increasingly broader context compared to its predecessor. As a result, the higher layers of the network are exposed to a progressively wider context.

Similar to Convolutional Neural Networks, the transformations within the same layer of a TDNN are tied across time. This temporal tying helps reduce the number of parameters and achieves transformation invariance to time shifts in the input, making these networks efficient and robust for sequence processing tasks. Historically, TDNNs are considered precursors to CNNs[14]. The model architecture is represented in the figure below.

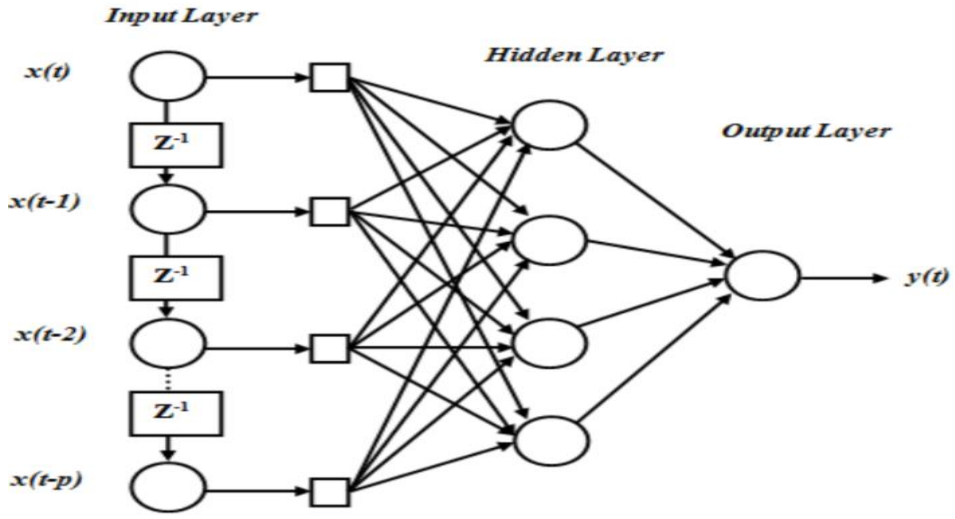


Figure 15: TDNN architecture[15]

2.2.3.5 Self-Attention

Attention mechanisms[16] in deep learning are inspired by human biology, where focus is directed towards distinctive parts when processing large volumes of information[17]. It is conceptualized as a vector of importance weights which prioritizes certain elements of the input data to enhance predictions [18]. Self-attention in audio processing is a technique used to capture relationships and dependencies within an audio sequence. It is particularly useful for tasks such as speech recognition, audio classification, and music generation. It allows a model to weigh the importance of different parts of an input sequence when making predictions. In the context of audio, this means the model can focus on various segments of an audio signal to capture temporal dependencies and patterns. The model architecture is represented in the figure below.

[9] What are Recurrent Neural Networks? | IBM (June 2024)

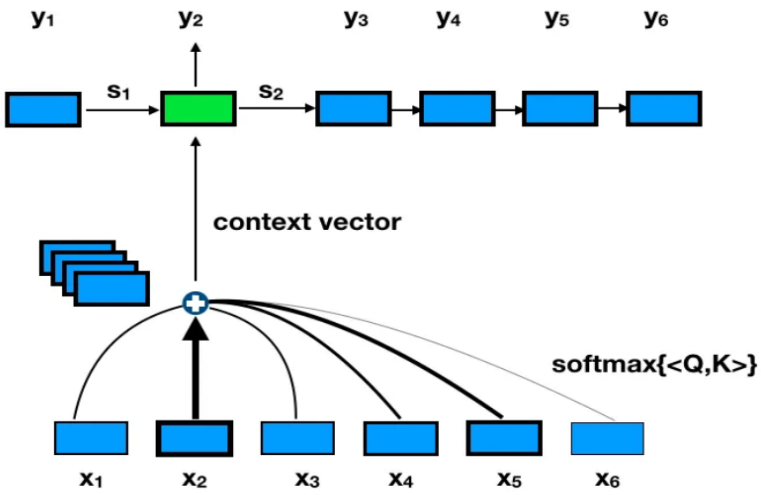


Figure 16: A simple Attention mechanism[10]

2.2.3.6 Autoencoder

Autoencoders[19] are neural networks tailored for dimensionality reduction, comprised of an encoder and a decoder. The encoder compresses input data, like audio signals, into a lower-dimensional representation using a weight matrix and activation functions such as sigmoid for non-linear mappings. The decoder then reconstructs the output from this compressed form, aiming to minimize the weighted reconstruction error through optimization techniques like stochastic gradient descent. This process not only adjusts weights until convergence but also enables autoencoders to learn efficient encodings automatically[20]. In the context of [19]audio classification, this capability is crucial as it enhances feature extraction for tasks such as noise reduction and audio signal processing, helping to isolate pertinent audio features and improve classification accuracy. The model architecture is represented in the figure below.

[10] Attention Mechanisms in Deep Learning — Not So Special | by Dr Stephen Odaibo | The Blog of RETINA-AI Health, Inc. | Medium

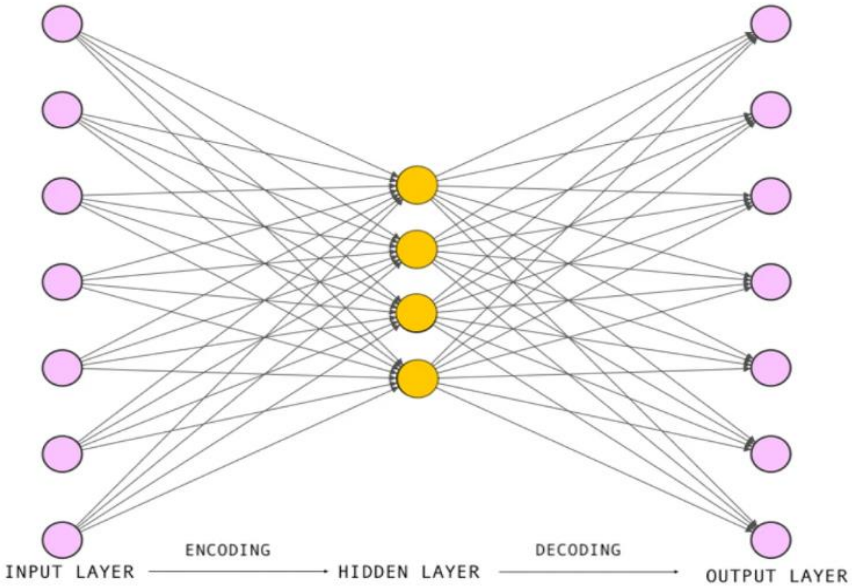


Figure 17: Architecture of general autoencoder[21]

2.2.3.7 Wavenet

Wavenet employs stacked dilated causal convolutional layers to model audio waveforms effectively. It defines the probability of waveforms through a product of conditional probabilities, optimizing its architecture for broader receptive fields via dilated convolutions. These convolutions allow it to cover more input within each layer, expanding its capacity without increasing the number of parameters significantly. wavenet also features gated activation units and employs both residual and skip connections to enhance training efficiency and support deeper network architectures, greatly improving the model's performance and speed of convergence[22]. The model architecture is represented in the figure below.

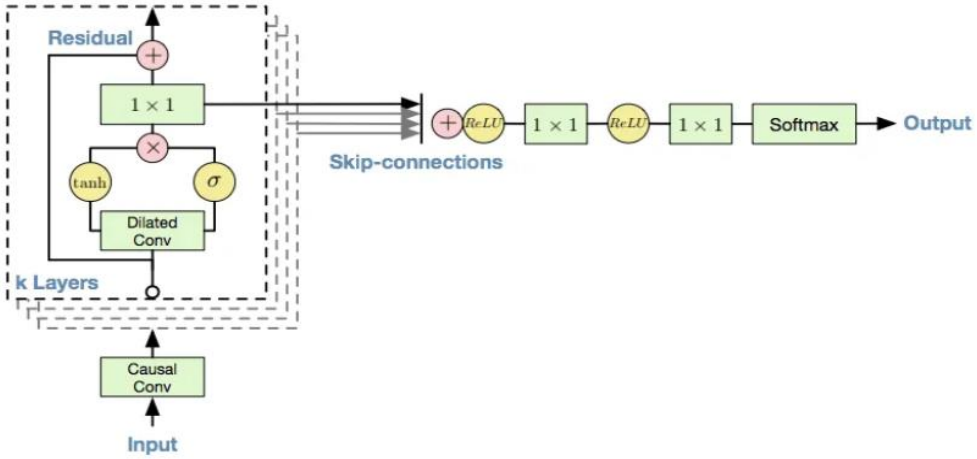


Figure 18: Overview of wavenet architecture[11]

[11] Understanding WaveNet architecture | by Satyam Kumar | Medium (June 2024)

2.2.4 Evaluation

to evaluate the performance of each deep learning model and understand their effectiveness, it is essential to measure their accuracy, F1 score, recall, and precision, where each metric provides a different type of insight into the strengths and weaknesses of a model.

2.2.4.1 Accuracy

It gives you the overall effectiveness of a classifier. Accuracy is most useful when each class label is equally important and the classes are balanced.

$$accuracy = \frac{Number\ of\ correct\ predictions}{total\ number\ of\ predictions} \quad 10$$

2.2.4.2 F1 Score

It is a better measure than accuracy for cases where the class distribution is imbalanced. The F1 score is especially useful when you need to balance precision and recall.

$$f1_{score} = 2 \times \frac{precision \times recall}{precision + recall} \quad 11$$

2.2.4.3 Recall

This measures the ability of a model to find all the relevant cases within a dataset. Specifically. High recall indicates that an algorithm returned most of the relevant results.

$$recall = \frac{true\ positives}{true\ positives + false\ negatives} \quad 12$$

2.2.4.4 Precision

This measures the accuracy of positive predictions; precision is a critical measure when the costs of false positives are high.

$$precision = \frac{true\ positives}{true\ positives + false\ positives} \quad 13$$

2.2.4.5 Confusion matrix

It is a table used to describe the performance of a classification model on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm. The matrix compares the actual target values with those predicted by the machine learning model, providing insight into the number of correct and incorrect predictions made by the model broken down by each class.

2.3 Part two

The application of machine learning and deep learning techniques to breathing sounds has shown great promise for the early and accurate detection of various respiratory diseases. Table 1 below shows the most relevant works:

num	Title	Published	Date of publishing	Algorithm used	evaluation	Resources
01	Computerized lung sound-based classification of asthma and chronic obstructive pulmonary disease (COPD)	Biocybernetics and Biomedical Engineering	17 December 2021	support vector machine (SVM), decision tree (DT), k-nearest neighbour (KNN) and discriminant analysis (DA)	accuracy of 99.3% using the decision tree (DT)	Dataset contains 80 asthma, 80 COPD and 80 healthy recordings
02	Automatic Covid-19 disease diagnosis using 1D convolutional neural network and augmentation with human respiratory sound based on parameters: cough, breath, and voice	AIMS Public Health.	2021 March 10	one dimension (1D) CNN	90% accuracy	'COVID-19 crowdsourced sounds data' dataset
03	Compressed Sensing Data with Performing Audio Signal Reconstruction for the Intelligent Classification of Chronic Respiratory Diseases	Advances in Artificial Intelligence for Biomedical Signal and Image Analysis	28 January 2023	a 17-layered 2D-convolutional neural network (CNN) with features of MFCC and spectrograms	accuracy of 92.30%	ICBHI 2017 dataset
04	Deep learning based respiratory sound analysis for detection of chronic obstructive pulmonary	PeerJ Computer Science	February 11, 2021	Convolutional Neural Network based deep learning methodologies	Score of 93%	Dataset contains recordings containing 6898 respiratory cycles, of which 1,864

	disease					contain crackles, 886 contain wheezes and 506 contain both crackles and wheezes
05	Abnormal respiratory sounds classification using deep CNN through artificial noise addition	Pulmonary Medicine	17 Novembe r 2021,	Convolutional neural network	Accuracy about 95%	Not mentioning about dataset used
06	Application of deep learning for bronchial asthma diagnostics using respiratory sound recordings	*PeerJ Computer Science*	2023-01-10	DenseNet201 model	The metrics for the classifier "asthmatic patient/non-asthmatic patient and healthy" were 92%, 82%, and 87%, respectively	An anonymous dataset is used containing 1238 records of respiratory diseases sounds of patient and 133 records of volunteers
07	Cough Sound Detection and Diagnosis Using Artificial Intelligence Techniques: Challenges and Opportunities	IEEE Journal of Biomedical and Health Informatics	15 July 2021	Artificial Neural Networks (ANN), Deep Neural Network (DNN), and Recurrent Neural Network (RNN). Also, the Support Vector Machine (SVM)	The ANN achieved 86% sensitivity, 91% specificity, 91% accuracy	They didn't mention specific dataset
08	Explainable attention ResNet18-based model for asthma detection using stethoscope	Engineering applications of artificial intelligence	Novembe r 2023	Attention network and machine learning technique (KNN or SVM)	accuracies of 99.19% and 99.73% using KNN and SVM classifiers, respectively.	dataset of lung sounds with 95 asthmatics and 108 healthy

	lung sounds					
09	Respiratory sound-base disease classification and characterizatio n with deep/machine learning techniques	Biomedical Signal Processing and Control	4 October 2023	supervised Machine learning and deep learning techniques	The model achieves an accuracy score of 0.998 and an F1-score of 0.998 for distinguishin g between healthy and respiratory disease-affected patients and an accuracy of 0.947 and an F1-score of 0.94 for categorizing 'unhealthy' records into COPD, asthma	Respiratory Sound Dataset, Coronohack Dataset, and Coswara Dataset
10	Deep Neural Network for Respiratory Sound Classification in Wearable Devices Enabled by Patient Specific Model Tuning	IEEE	3 JUNE 2020	CNN-RNN model using Mel-spectrograms	hybrid CNN-RNN model achieves a score of 66.31%	ICBHI 2917

Table 1: Most relevant works

2.3.1 Machine learning

Machine learning techniques include traditional ones are widely used for classification, regression and clustering tasks. For instance.[23], a study on lung sound-based classification uses support vector machines (SVM), decision trees (DT), k-nearest neighbours (KNN) and discriminant analysis to differentiate asthma from COPD and healthy lung sounds. Utilizing a dataset comprising 80 asthma, 80 COPD and 80 healthy audio samples, the study achieved remarkable precision reaching 99.3% accuracy using the Decision Tree model [24].

2.3.2 Deep learning

Deep learning is a subset of machine learning, unlike the ML techniques that require manual feature extraction and pre-processing, DL automates these processes by learning high-level features from data incrementally which makes it more effective.

A commonly used model in this domain is the Convolutional Neural Network (CNN), which has proven highly effective for detecting respiratory anomalies. For example, the application of a 1D CNN has been successful in identifying COVID-19 by analysing human respiratory sounds using 'COVID-19 crowdsourced sounds data' dataset which includes 121 cough sounds and 94 non-cough sounds resulted in an impressive 90% accuracy in identifying COVID-19 based on cough, breath, and voice analyses [25]. Furthermore, CNNs have been extensively utilized for the early detection of Chronic Obstructive Pulmonary Disease. One study employed a 17-layered 2D-CNN that incorporates features such as MFCC and spectrograms, trained on the ICBHI[12] dataset that contains different pulmonary diseases including COPD, achieving an accuracy of 92.30% [26]. Another study implemented a more complex CNN architecture, consisting of input layers, convolutional 2D layers, dropout layers, max pooling 2D layers, and a dense layer, trained on a dataset comprising 6898 respiratory cycles including 1864 with crackles, 886 with wheezes and 506 with both reaching an accuracy of 93%[27]. Additionally, a different study aimed to enhance the identification of seven types of respiratory sounds by combining Fourier analysis and artificial noise addition with deep CNNs, although it does not provide extensive details about the dataset used, achieving an accuracy of 95%[28].

For of asthma detection, the DenseNet201 model has been used achieving results of 92%, 82%, and 87% accuracy respectively, this model is trained on a dataset containing 1238 records of respiratory sounds from patients and 133 records from healthy individuals, focusing on binary classification of asthmatic or non-asthmatic states during calm breathing[29]. Another model is Respire-Net which utilizes CNNs and incorporates innovative techniques to enhance the accuracy of classifying respiratory sounds into four categories. This model is tested using the ICBHI dataset achieving impressive results[30].

For further precision, a variety of models including Artificial Neural Networks, Deep Neural Networks and Recurrent Neural Networks have been applied to cough sound detection and diagnosis. This diverse toolkit addresses various aspects of the data processing pipeline from feature extraction to classification. Notably, the best-performing model, an ANN that achieved 86% sensitivity, 91% specificity, and 91% accuracy[31].

2.3.3 Combined

The combined approaches exploit the deep learning strengths of machine learning, using deep learning to advance features and machine learning for precise classification, this integration often leads to improved accuracy and interpretability

[12] https://bhichallenge.med.auth.gr/ICBHI_2017_Challenge.

qualities essential for clinical applications.
One effective integration employs attention mechanisms alongside traditional machine learning algorithms like SVM and KNN for asthma detection. Utilizing data collected via a smart stethoscope which included samples from 95 asthmatics and 108 healthy individuals, this method achieved impressive accuracies of 99.19% with KNN and 99.73% with SVM classifier [32] s. Another study used a two-step process whose first step involved using machine learning to extract feature vectors from breathing sounds for an initial health diagnosis while the second step differentiated several respiratory diseases, including chronic obstructive pulmonary disease utilizing three datasets (Respiratory Sound Dataset, Coronohack Dataset, and Coswara Dataset) to achieve an overall accuracy of approximately 99.8% [33].
Furthermore, some research explores the synergy between different deep learning models. For instance, a hybrid model that combines convolutional networks (CNNs) with recurrent neural networks (RNNs) is developed, employing Mel spectrograms for the classification of sounds in wearable devices. This model employs CNNs to extract spatial features and RNNs to analyse temporal dependencies, tested on the ICBHI dataset which resulted in a performance score of 66.31%[34].

2.4 Conclusion

The chapter cover a discussion of the use of deep learning and machine learning techniques for analysing breathing sounds, highlighting the improvements and various approaches considered by researchers to enhance the diagnostic efficiency and reliability of breathing sound analysis. As well as, it provides an overview of key steps in audio pre-processing, including segmentation, data augmentation, feature extraction, and addressing the imbalance issue. Additionally, covering various deep learning models such as CNN, LSTM, RNN, TDNN, autoencoder and wavenet, and explain how to evaluate their performance.

Chapter 03: ***Proposed approaches***

3.1 Introduction

In recent years there has been a growing interest in applying AI to audio-based medical diagnostics. This study focuses on detecting respiratory diseases using various deep learning models including individual and combined approaches. The goal is to determine which model performs best.

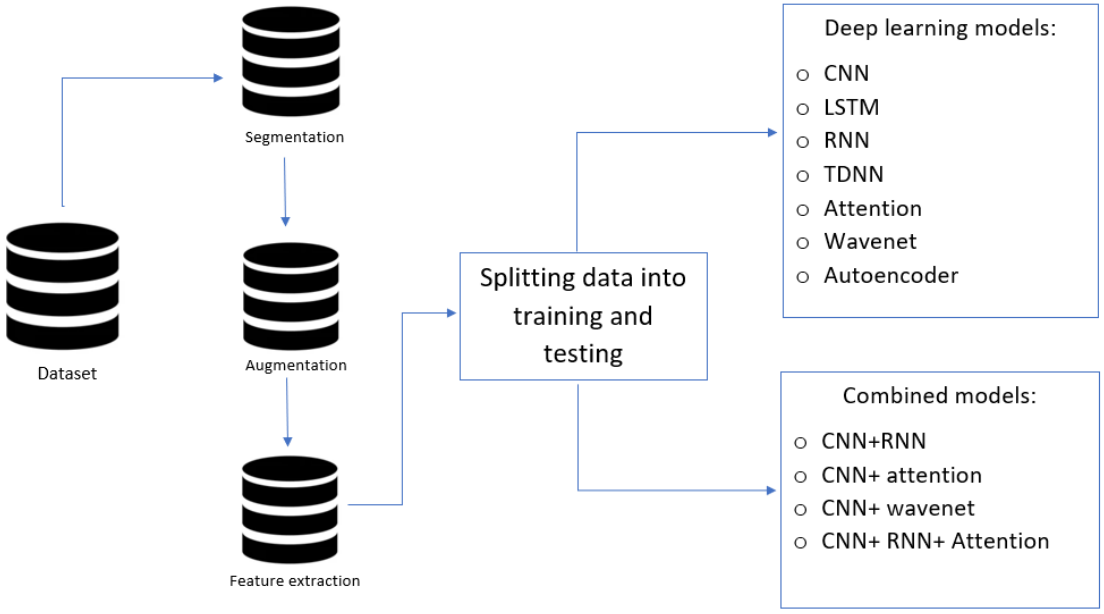


Figure 19: workflow for processing audio data to detect respiratory disorders using deep learning models.

3.2 The dataset

A dataset under the name of 'A dataset of lung sounds recorded from the chest wall using an electronic stethoscope'[13] is used, that comprises approximately 336 audio files in (.wav) format. The names of the audio files are structured to encode several pieces of information, providing context and details for each recording:

- Filter type (B: BELL 20-200Hz, Diaphragm 100-500 Hz, Extended range 50-500 Hz).
- Patient number: P1-P112.
- the diagnosis (Asthma, COPD, Bron, Pneumonia, Pleural effusion, heart failure, N: healthy, Lung fibrosis), it can be more than one label
- the sound type (Inspiratory: I, Expiratory: E, Wheezes: W, Crackles: C, N: Normal)
- the location on the chest wall from which the recording was taken (Posterior: P Lower: L Left: L Right R, UPPER: U, ANTERIOR: A, MIDDLE: M).
- Age
- Sex (F: female, M: male)

[13] https://data.mendeley.com/datasets/jwyy9np4gv/3

Example file name: ‘BP8_Pleural Effusion,ICB,PLL,70,M’ indicates:

- A Bell filter was used.
- Patient number 8.
- Diagnoses include Pleural Effusion.
- Contains Inspiratory and Crackles sounds.
- Recorded on the Posterior Lower Left of the chest.
- The patient is a 70-year-old male.

In this study, the diagnoses is used as labels for classifying the audio files. Where the labels are as shown in the figures below

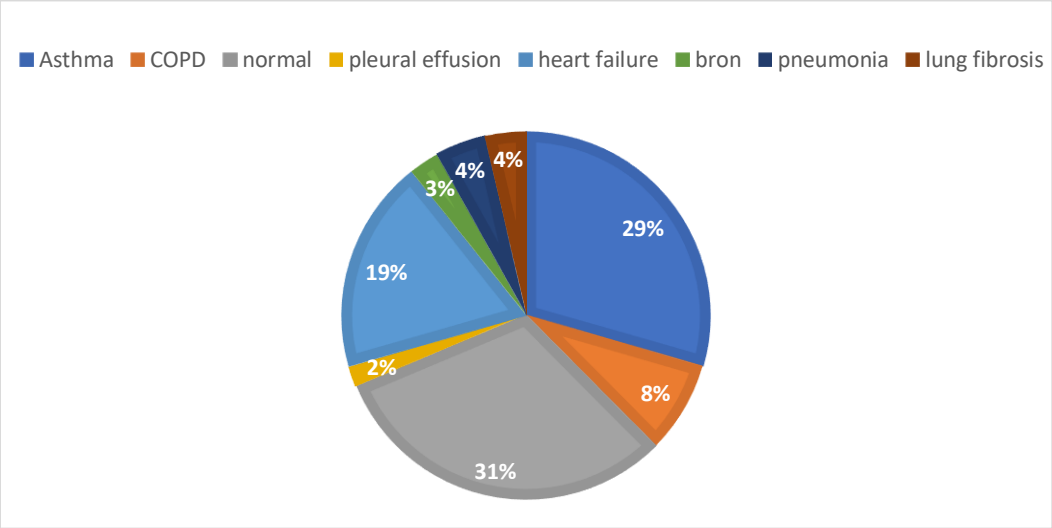


Figure 20:number of classes in the dataset

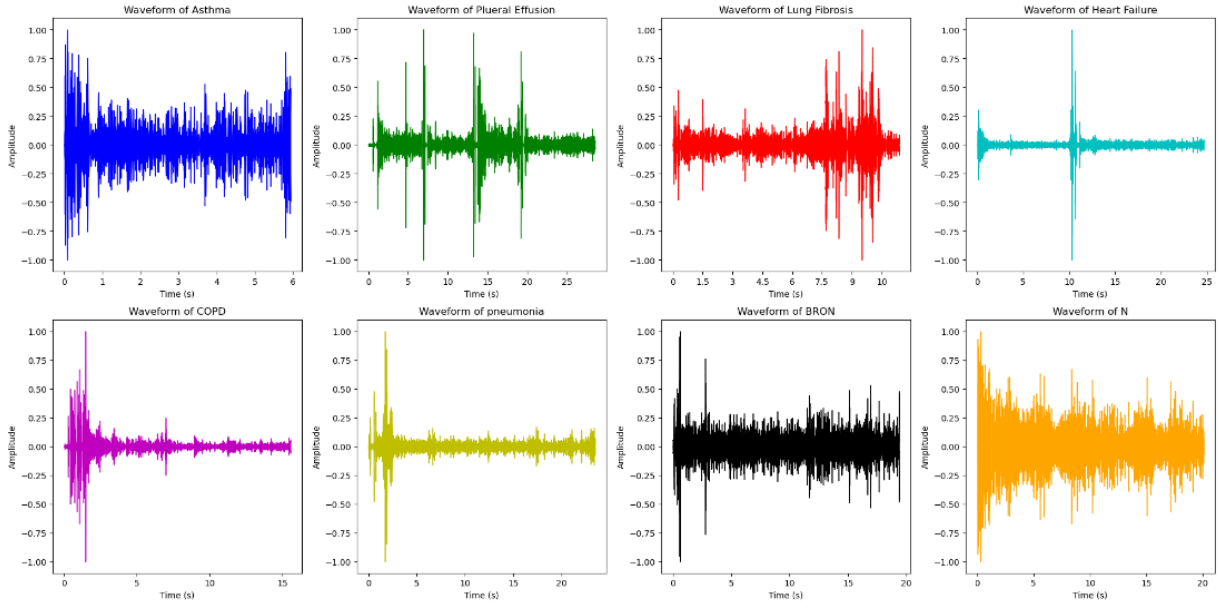


Figure 21: visualisation of each label

3.3 Experimentations steps

3.3.1 Segmentation

It is an important step in pre-processing of audio data, two main functions are used:

- Noise reduction

  This function takes as parameters the audio, the number of samples of audio carried per second, measured in Hz or kHz and the noise duration that specifies the duration of audio in milliseconds at the beginning of the track that is assumed to be noise.

a. Noise profile estimation: The process additionally starts with using the noise duration of the audio to estimate the noise profile, this is based on the assumption that this initial segment primarily contains noise, which is common in recordings. Then calculates MFCCs for noise segments and the entire audio file.
b. MFCC calculation: Calculates the MFCCS for the noise segment and the entire audio file.
c. Noise reduction: noise profile is estimated by calculating the average of the MFCCs over the noise segment in order to reduce the noise components present in the overall audio.
d. Audio Reconstruction: after noise reduction, the audio signal is reconstructed from the modified MFCCs, the reconstruction involves transforming the MFCCs back into a waveform.

- Audio segmentation

  This function takes as parameters the audio, the number of samples of audio

a. Frame division: The audio is divided into frames using a ‘frame_length’ of 2048 samples using larger frame_length means checking more of the audio at once which leads to a better overall picture of what's happening in that larger section but with less detail about the smaller variations, and a ‘hop_length’ that determines how far to move forward in the audio after measuring one section before measuring the next using 512 samples, a smaller hop_length makes the measurement smoother and more detailed. These values typically balance between computational efficiency and the granularity of the analysis
b. Energy calculation: Energy is computed for each frame as the sum of squared amplitudes, providing a measure of the frame's loudness or intensity, then normalized in order to help to scale the energy values between 0 and 1, facilitating the application of a uniform threshold.
c. Segmentation: A threshold (thresh = 0.1) is used to detect significant audio segments. Frames with normalized energy above this threshold indicate the presence of important audio content, whereas lower energy suggests silence or background noise.

Before implementing segmentation, they were 336 audio files, after segmentation the number increased to 1,604 audio files.

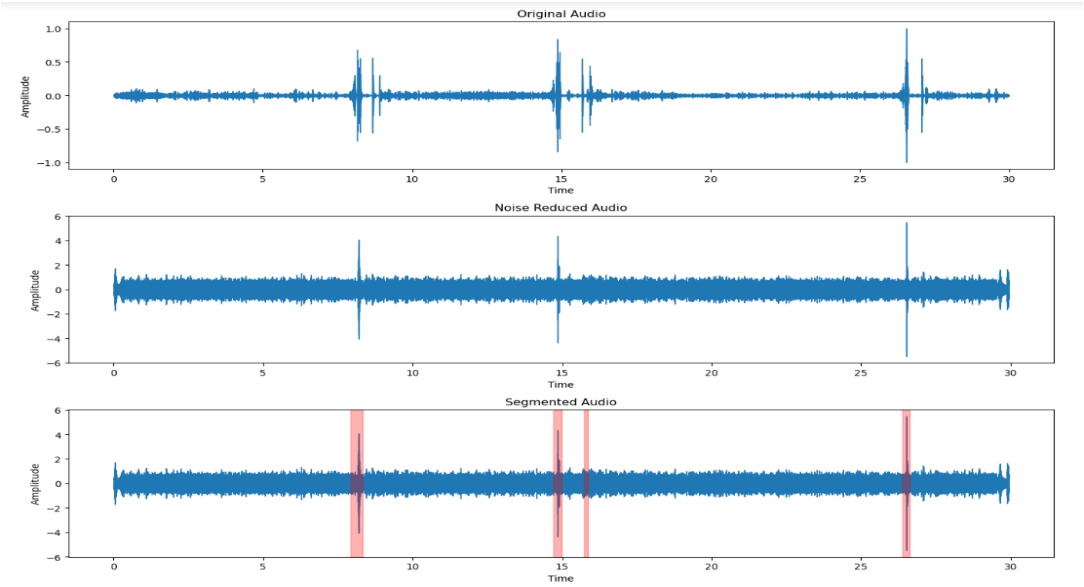


Figure 22: audio after implementing noise reduction and segmentation

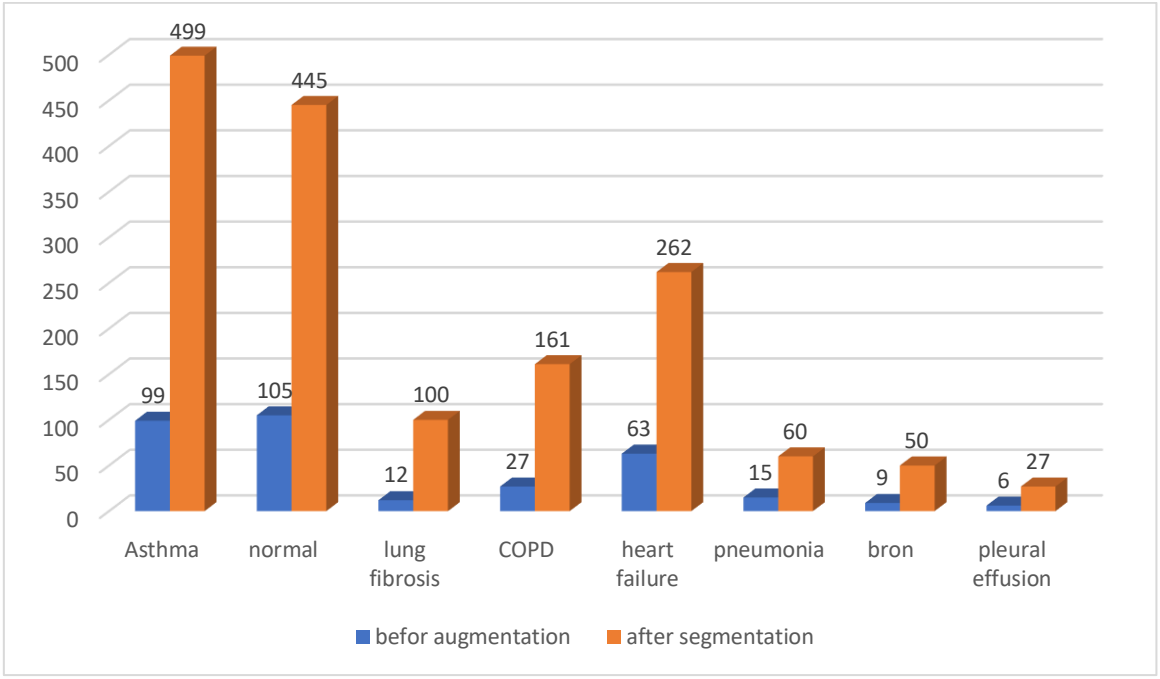


Figure 23:number of classes before and after segmentation

3.3.2 Data augmentation

Before applying augmentation techniques, the audios are normalized by dividing the entire audio signal by its maximum absolute amplitude to ensure that the waveform amplitude ranges between -1 and 1. Following normalization, several and combined techniques are used in order to augment the audios without losing the quality.

- Voice adjustment and noise

  Volume adjustment is an obvious concept where the audio signal's amplitude is scaled by a factor where a predefined list is used [0.9, 1.1, 1.2]. For example, a factor of 0.9 decreases the volume to 90% of its normalized level whereas a factor

of 1.2 increases it to 120%. This adjustment is achieved by multiplying each audio sample by the respective volume factor. To further boost the data, the Gaussian noise is introduced to each audio sample. This noise follows a probability density function typical of a normal distribution and is implemented with very low parameters to ensure that the quality of the respiratory sounds stays intact while still simulating real-world noise.

After applying these augmentation techniques to the segmented audios, the number of audio files increased significantly resulting in a total of 9624 augmented audio files.

- Time shifting and dynamic range compression

  These augmentations are applied sequentially, initially starting with time shifting where it moves the audio signal forward or backward, the forward time shifting is used which is preferred over backward one for its preservation of original audio. The function calculates how many samples to shift the audio forward based on the simple rate making sure that it does not exceed the audio length, then the shift will be applied by appending zeros at the beginning of the audio array and truncate the same number of samples from the end of the audio to keep the original length.

  Following time shifting, the audio is converted to a 16-bit integer format where values range from -32768 to 32767. This conversion is necessary for dynamic range compression which then compresses the audio by reducing the volume of loud sounds and amplifying quieter sounds. At the end of the process, the audio is converted back to the original floating-point format to facilitate further digital processing.

  After applying these augmentation techniques to the segmented audios, the number of audio files increased significantly resulting in a total of 1604 augmented audio files.

- Mild pitch shifting

  It is a digital signal processing technique that changes the pitch of an audio signal without changing the duration of the original audio.

  this technique is implemented on normalized segmented audios where each audio is randomly pitch-shifted within an interval of -2 to +2 semitones. This approach ensures the pitch variations are subtle but effective, providing a realistic range of pitch adjustments that enhance the diversity of the dataset without distorting the audio's original characteristics.

  After applying mild pitch shifting technique to the segmented audios, the number of audio files increased significantly resulting in a total of 1604 augmented audio files.

- Equalization

  After normalizing the segmented audio files, equalization technique is implemented, which consist of three primary steps.

  The process begins with applying a Short-Time Fourier Transform (STFT) to the audio data. This method divides the audio into short frames and computes the STFT for each frame separately. This transformation shifts the audio from the time domain into the frequency domain, allowing for frequency-specific manipulation.

The second step involves the application of a linearly increasing gain filter. The values of this filter are spaced between 1 and 2; it starts at 1 for the lowest frequencies and increases linearly to 2 for the highest frequencies. This method selectively enhances higher frequencies more than the lower ones, effectively brightening the audio signal.

As the final step, the modified frequency-domain data is converted back into the time domain using an Inverse STFT (ISTFT). This reconstruction maintains the original number of segmented audios while modifying their frequency characteristics through the applied equalization.

After applying equalization technique to the segmented audios, the number of audio files increased significantly resulting in a total of 1604 augmented audio files.

After completing the audio augmentation process using multiple techniques where each technique is used in a way that kept the original quality of the audio while adding variety to its sound. As a result, the number of audio files in the dataset increased dramatically to 14427 as shown in the table below.

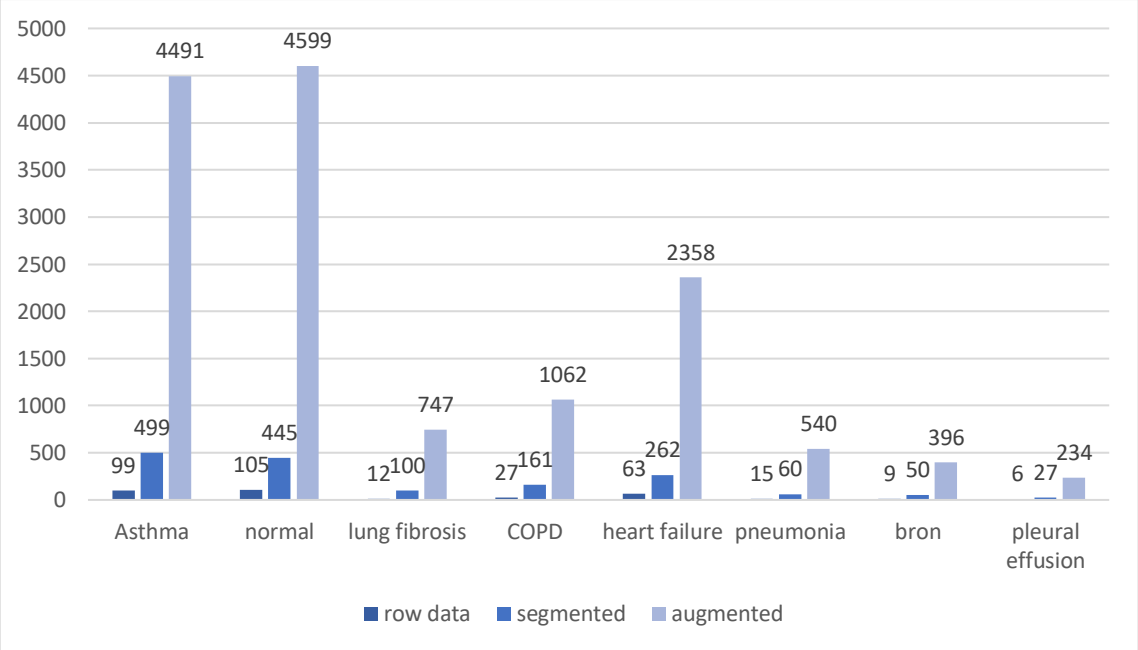


Figure 24: number of classes during pre-processing

3.3.3 Feature extraction

- MFCCs extraction

  This function takes as parameters: the audio signal as a numpy array, The number of samples per second in the audio file and the number of MFCC values to be calculated for each frame of audio taking it as 13 because it captures enough of the perceptually relevant aspects of the audio spectrum.

  The process additionally, a pre-emphasis filter is applied to the signal to amplify the high frequencies, then splitting the audio signal into short frames typically around 20-40 ms long, where each frame is multiplied by a windowing function to

minimize the discontinuities at the beginning and end of each frame. A FFT is applied to convert each frame from frame domain to frequency domain, this frequency scale is converted later to the Mal scale through a set of filters known as Mel filterbank and apply the logarithm on it. Finally, a DCT applied to the log Mel spectrum to de-correlate the filter bank coefficients and yield a compressed representation of the filter banks. The first 13 coefficients are kept as they contain the majority of the information relevant to the perceived sound.

After extracting the MFCCs for each frame, the next step is to average these coefficients across all frames. This averaging reduces the data to a single mean vector for each MFCC coefficient across the entire audio file.

- Spectrogram extraction

  This function takes as parameters: the audio as a numpy array, the sample rate, the number of points used in the Fourier Transform window during the computation of the STFT, taking it as 2048 which allows for detailed frequency analysis and the hop_length (520) which is the number of samples between the starts of two consecutive frames when the STFT is computed, so the time step between successive frames is approximately 11.6 millisecond.

  Additionally, The STFT is applied to the audio data to transform it from the time domain to the frequency-time domain. The STFT operates by slicing the continuous audio signal into short frames and applying the Fourier Transform to each frame, The output of the STFT is a complex matrix where the magnitude of each element represents the amplitude of a particular frequency at a certain time, and the angle represents the phase. The magnitude of this complex number is squared to convert it from amplitude to power. Finally, the power spectrogram is converted into a logarithmic scale.

- Combined

  At the beginning, each one is implemented individually then combine them by concatenating the mean of the spectrogram features along with the MFCC features into a single feature vector per audio file and saving them as '.npy' files, This combination can be particularly powerful as MFCCs capture the melodic content well while the spectrogram captures the intensity and texture of the sound, offering a multidimensional perspective of the audio's characteristics.

3.3.4 Data preparation

- Label extractions: labels are extracted directly from name of the audios then using 'LabelEncoder' for Transforming the text labels into a numerical format
- Converting any NaN value to 0 and making sure that the dataset doesn't contain any missing values and ensures all feature data is of type float32.
- Solving the problem of the imbalance, over sampling technique is used by duplicating random instances in the minority classes.
- Split the data into training and testing sets, with the testing set comprising 20% of the dataset.

3.4 Models

All the models are compiled with the Adam optimizer and sparse categorical crossentropy loss that does fit with multi-class classification tasks where each class

label is provided as a single integer, and the training of the models includes: from 10 to 30 epochs, 32 batch size and early stopping that stops training if the validation loss doesn't improve for 3 consecutive epochs.

3.4.1 Deep learning models

3.4.1.1 Convolutional neural network (CNN)

- Model creation: to build a sequential model using several layers configured for processing 1D sequence data:
  - Conv1D layers: 2 layers
  - First one with 32 filters and 3 as a kernel size
  - Second one with 64 layers and a kernel size of 3
  - MaxPooling1D layers: 2 layers
  - Both layers use a pool size of 2
  - Dropout layers: 2 layers
  - Botth layers use a dropout rate of 0.2
  - Flatten layer: a layer
  - Converts the output of convolutional and pooling layers into a 1D array.
  - Dense layers: 2 layers
  - First layer with 100 neurons and 'ReLu' activation function.
  - Second one which is the output layer with 8 neurons and 'softmax' activation function
- Data reshaping: It does reshape the input data to fit the model's input requirements, it transforms the data to have an additional dimension which represents channels.
  The table bellow shows the history of the model

Layer	*Output shape*	*Param*
Conv1D	(None,1036,32)	128
MaxPooling1D	(None,518,32)	0
Dropout	(None,518,32)	0
Conv1D	(None,516,64)	6208
MaxPooling1D	(None,258,64)	0
Dropout	(None,258,64)	0
Flatten	(None,16512)	0
Dense	(None,100)	1651300
Dense	(None,8)	808
Total params: 1658444 *Trainable params: 1658444* *Non-trainable params: 0*		

Table 2: Summery of the CNN model

3.4.1.2 Long Short-term memory (LSTM)

- Model creation: To build the neural model:
  - Bidirectional LSTM layers: 2 layers

- The first one with 64 units, it does process the input data in both forward and reverse directions capturing patterns that may be missed when the data is processed in only one direction.
- The second one with 32 units that prepare the output for the dense layer.

- Batch normalization layers:2 layers are used that normalize the activation of the previous layer where each batch applies a transformation that maintains the mean output close to 0 and the output standard deviation close to 1.
- Dropout layers: 2 layers are used that randomly set a fraction 20% of input units to 0 at each update during training time which prevent overfitting.
- Dense layers: 2 layers

- The first one has 100 neurons with ReLu activation, providing a non-linear transformation to the learned features.
- The second one has 8 neurons with softmax activation suitable for multi-class classification.

- Data reshaping: Reshaping the input data to fit the LSTM's input format by adding a channel dimension.

The table bellow shows the history of the model

Layer	*Output shape*	*Param*
Bidirectional	(None,1038,128)	33782
Batch_normalization	(None,1038,128)	512
Dropout	(None,1038,128)	0
Bidirectional	(None,64)	41216
Batch_normalization	(None,64)	256
Dropout	(None,64)	0
Dense	(None,100)	6500
Dense	(None,8)	808
Total params: 83084 *Trainable params: 82700* *Non-trainable params: 384*		

Table 3:Summery of LSTM model

3.4.1.3 Recurrent Neural Network (RNN)

- Model creation: Defining a sequential model with a total of 8 layers:

- GRU layers: 2 layers:

- The first one with 64 units and returns sequences to feed the next layer.
- The second one has 32 units and doesn't return sequences.

- Dropout layers: 2 layers where each set at 0.3, these layers randomly set a portion of input units to 0 at each update during training helping to prevent overfitting.
- Batch normalization: 2 layers that normalize the activations of the previous layers at each batch keeping a mean output close to 0 and the output standard deviation close to 1.
- Dense layers: 2 layers where:

- the first one has 100 neurons using ReLu activation providing non-linearity to the learning process.
- then, the second one that has 8 neurons using softmax activation.

- Data reshaping: the data are reshaped to fit the RNN input requirements by adding an additional dimension to represent channels.
  The table bellow shows the history of the model

Layer	*Output shape*	*Param*
GRU	(None,1038,64)	12864
Dropout	(None,1038,64)	0
Batch_normalization	(None,1038,64)	256
GRU	(None,32)	9408
Dropout	(None,32)	0
Batch_normalization	(None,32)	128
Dense	(None,100)	3300
Dense	(None,8)	808
Total params:26764 *Trainable params:26572* *Non-trainable params: 192*		

Table 4: summery of the RNN model

3.4.1.4 Time-Delay Neural network (TDNN)

- Model creation: Defining a sequential model tailored for 1D sequence data that includes:
  - Conv1D layer: 3 layers where each layer uses the ReLu activation function, the dilation rate increases with each subsequent layer allowing the network to process information from the input sequence over larger timescales without increasing computational complexity.
  - Dropout layers: 3 layers, each with 0,2 as a dropout rate, these layers help to prevent overfitting by randomly setting a proportion of input features to zero.
  - Batch normalization:
  - Flatten layer: one layer that does flatten the output of the last Conv1D layer to form a 1D vector to be fed into the dense layers.
  - Dense layers: 2 layers:

  - The first one with 100 neurons.
  - the output layer with 8 neurons and uses softmax activation.
- Data reshaping: the data is reshaped TDNN model that requires a third dimension representing channels.
  The table bellow shows the history of the model

Layer	*Output shape*	*Param*
Conv1D	(None,1036,32)	128
Dropout	(None,1036,32)	0

Conv1D	(None,1032,64)	6208
Dropout	(None,1032,64)	0
Conv1D	(None,1024,128)	24704
Dropout	(None,1024,128)	0
flatten	(None,131072)	0
Dense	(None,100)	13107300
Dense	(None,8)	808
Total params:13139148 *Trainable params:13139148* *Non-trainable params: 0*		

Table 5: TDNN model

3.4.1.5 Attention

Transformer architecture is used as a part of the attention model

- Transformer Encoder: This function constructs a transformer block:
  - Normalization layer: that makes the training stable by normalizing the activations within the layer.
  - MultiHead Attention: It does allow the model to focus on different parts of the input sequence simultaneously.
  - Dense layers: There are 2 dense layers fully connected that process the output further.
  - Dropout: it is used for regularization to prevent overfitting by randomly setting a fraction of the input units to zero at each update during training time.
  - Skip connections: These are used to add the input directly to the output of the attention and dense layers, which helps mitigate the vanishing gradient problem in deep networks.
- Model creation:
  - Input layer: Where the shape of the input data is defined.
  - Transformer Encoder: To apply the Transformer block to the input.
  - Flatten layers: Using 2 layers:

  - The first one is hidden with 100 neurons and ReLu activation.
  - The output layer with softmax activation.
- Data reshaping: the input data is reshaped to fit the model by adding extra dimension.

  The table bellow shows the history of the model

Layer	*Output shape*	*Param*
Input	(None,1038,1)	0
Normalization_layer	(None,1038,1)	2
MultiHead_attention	(None,1038,1)	1793
Dropout	(None,1038,1)	0
Tf.opertators.add(TFOpLambda)	(None,1038,1)	0
Normalization_layer	(None,1038,1)	2

Dense	(None,1038,64)	128
Dropout	(None,1038,64)	0
Dense	(None,1038,1)	65
Tf.opertators.add(TFOpLambda)	(None,1038,1)	0
flatten	(None,1038)	0
Dense	(None,100)	103900
Dense	(None,8)	808
Total params:106698 *Trainable params: 106698* *Non-trainable params: 0*		

Table 6: summery of attention model

3.4.1.6 Autoencoder

- Model creation:
  - Input layer: Takes input with the shape defined.
  - Autoencoder:

  - Encoder: that contains 2 dense layers with ReLu activation to compress the input into a lower-dimensional representation, The first one with 128 neurons while the second one has 64 neurons.

  - Decoder: contains as well 2 dense layers two dense layers that attempt to reconstruct the original input from the encoded representation. The first one with 128 neurons using ReLu activation, while the second one uses sigmoid activation.
  - Classifier: Takes the output from the last encoder layer and flattens it (flatten layer) then passes it through a dense layer with softmax activation to produce class probabilities for classification tasks (dense layer).
  - Outputs: The model has 2 outputs one from the decoder and another from the classifier.

  The model is compiled with Adam optimizer and mean squared error (MSE) loss
- Data reshape: The data is reshaped to be flat for processing by the dense layers.
  The table bellow shows the history of the model

Layer	*Output shape*	*Param*
Input	(None,1038)	0
Dense	(None,128)	132992
Dense	(None,64)	8256
Dense	(None,128)	8320
flatten	(None,64)	0
Decoder_output (Dense)	(None,1038)	133902
Classifier_output(Dense)	(None,8)	520
Total params:283990 *Trainable params:283990* *Non-trainable params: 0*		

Table 7: Autoencoder summery

3.4.1.7 WaveNet

- Model creation: it includes
  - Input layer: Defines the shape of the input data.
  - Residual blocks: 3 are used, two parallel Conv1D layers with ‘tanh’ and ‘sigmoid’ activations known as ‘gated activation units’ that use dilated convolutions which allow the network to have a large receptive, then the outputs of these layers are multiplied element-wise, and the result passes through another Conv1D layer to produce a skip connection output.
  - Conv1D layer: For additional processing using ReLu activation.
  - Dropout layer: For regularization.
  - Flatten layer: To prepare the data for classification
  - Dense layer: An output layer with softmax activation for classification.
- Data reshaping: In order to fit the convolutional operations, the data is reshaped to include a channel dimension.

  The table bellow shows the history of the model

Layer	*Output shape*	*Param*
Input	(None,1038,1)	0
Conv1D	(None,1038,64)	192
Conv1D	(None,1038,64)	192
Tf.math.multiply(TFOpLambda)	(None,1038,64)	0
Conv1D	(None,1038,64)	4160
ADD	(None,1038,64)	0
Conv1D	(None,1038,64)	8256
Conv1D	(None,1038,64)	8256
Tf.math.multiply(TFOpLambda)	(None,1038,64)	0
Conv1D	(None,1038,64)	4160
ADD	(None,1038,64)	0
Conv1D	(None, 1038, 64)	8256
Conv1D	(None, 1038, 64)	8256
Tf.math.multiply(TFOpLambda)	(None, 1038, 64)	0
Conv1D	(None, 1038, 64)	4160
ADD	(None, 1038, 64)	0
Activation	(None, 1038, 64)	0
Conv1D	(None, 1038, 64)	4160
Dropout	(None, 1038, 64)	0
Flatten	(None,66432)	0
Dense	(None,8)	531464
Total params:581512 *Trainable params: 581512* *Non-trainable params: 0*		

Table 8: Wavenet summery

3.4.2 Combined models

In order to enhance the performance, we tried to combine some models:

3.4.2.1 CNN combined with RNN

It is one of the most common combination:

- CNN for feature extraction:
  - Conv1D layers: Using 2 layers:

  - First one with 32 filters and 3 as a kernel size with ReLu activation, this is the primary feature extractor from the input sequence.
  - Second one with 64 filters with the same kernel size and activation, capturing more complex features from the input.
    - MaxPooling1D: Using 2 layers that are used to reduce the dimensionality of the sequence which helps to make the representation more manageable and robust to small variations in the position of features within the sequence.
    - Dropout layers: Included after each pooling step to help prevent overfitting by randomly deleting units from the feature maps during training.
- Transition to RNN: The final CNN layer outputs a sequence where each element is a reduced and highly processed representation of the input, capturing important local features within segments of the input data, this output is directly fed into the RNN layers. In the provided model, the sequence output from the CNN layers is not flattened before it is fed into the RNN, allowing the RNN to process the sequential information where each timestep contains extracted features from the CNN.
- RNN for sequence modelling:
  - GRU layers: Using 2 layers:

  - The first one has 64 units and returns sequences feeding it to the next layer.
  - The second one has 32 units and does not return sequences, thus reducing the output to a single vector.
    - Batch normalization: 2 layers that normalize of the previous layer at each batch by applying a transformation that maintains the mean output close to 0 and the output standard deviation close to 1.
    - Dense layers:

  - Flatten: Flattens the output of the recurrent layers to a vector that can be fed into fully connected layers.
  - Dense layer: To interpret the features extracted by the convolutional and recurrent layers used a fully connected layer with 100 neurons and ReLu activation function.
  - Output layer: A dense layer with a softmax activation that outputs probabilities for each class in a classification task.
- Data reshape: the data is reshaped to include a channel dimension in order to fit the CNN input.

3.4.2.2 CNN combined with self-attention

- CNN layers:

- Conv1D layers: these layers extract features from the input data.
- MaxPooling1D: Reduces the dimensionality of the data.
- Dropout: It ignores randomly units during training.

- Transition to transformer: After processing data through CNN layers, it will be fed into the transformer encoder.
- Transformer encoder:
  - Normalization layer: It normalize the data within each feature.
  - MultiHeadAttention: It applies the attention mechanisms on the inputs.
  - Dropout: it does same work as the dropout layer in the CNN.
  - Residual connections: it does add the original input to the output of the attention mechanism.
  - Feed-forward network: Consists of two dense layers with a ReLu activation in between.
  - Output: Another skip connection adds the result of the feed-forward network back to the combined input and attention output providing another layer of preservation for the input signal through the network.
- Output layers:
  - Flatten: converts the multi-dimensional output of the transformer encoder into a 1D array suitable for input into fully connected layers.
  - Dense layers: using 2 layers:

  - The first one with 100 neurons uses ReLu activation.
  - Last layer uses the softmax activation to output probabilities for each class
- Data reshape: The data is reshaped to add a channel dimension to fit the Conv1D layers.

3.4.2.3 CNN combined with Wavenet

- Model creation:
  - Initial CNN layer: A 1D convolution layer followed by max pooling and dropout is used to extract initial features from the input data.
  - Preparation for Residual blocks: A 1D convolution adjusts the number of filters to match that used in the residual blocks.
  - Wavenet blocks: Multiple residual-blocks each one contains:

  - Two parallel convolutional layers with ‘tanh_out’ and ’sigmoid’ activations known as ‘gated activation unit’.
  - The outputs of these layers are multiplied element-wise (gated) to control the flow of information.
  - A skip connection sums the output of a 1x1 convolution (skip) on the gated output with the block's input.
  - Output layers: After the WaveNet blocks, the features are passed through additional convolutional, dropout, and flattening layers to prepare for classification. The final output is a dense layer with softmax activation.
- Data reshape: The training and test data are reshaped to fit the input requirements of the 1D convolution layers by adding an extra dimension for channels.

3.4.2.4 CNN combined with RNN and attention

- Model creation:
  - CNN layers: Initial layers that process the input sequences to extract features, using convolutional layers followed by max pooling and dropout for dimensionality reduction and regularization.
  - RNN layers: Process sequence data to capture temporal dependencies. GRUs are used for their efficiency and effectiveness in many sequence modelling tasks. Batch normalization is applied to stabilize and accelerate training.
  - Transformer layer: Containing:

  - Normalization layer
  - MultiHeadAttention
  - Feed-Forward network
  - skip connections
  -Dropout

  - Dense layers: A fully connected layer to interpret the features extracted and processed by the CNN, RNN and Transformer layers followed by the output layer which uses softmax activation,
- Data reshape: The data is reshaped to fit the model's input requirements, essentially adding a channel dimension.

3.5 Conclusion

In this chapter, every step utilized in the project is detailed, beginning with the pre-processing step, which is crucial for the effective training of models. Several deep learning models are experimented with aiming to achieve outstanding results.

Chapter 04: ***Results and discussion***

4.1 Introduction

This chapter presents the results of the performance evaluation of various deep learning approaches applied to respiratory disorder detection using audio recordings. It provides a comprehensive analysis, including discussions and comparisons of individual models such as CNN, LSTM, RNN, TDNN, WaveNet, and models incorporating attention mechanisms and autoencoders. Additionally, it evaluates the effectiveness of combined models that leverage the strengths of different architectures to achieve superior performance.

The evaluation focuses on key performance metrics, including accuracy, precision, recall, and F1 score, across different stages of data processing, including raw, augmented, and segmented data. The chapter aims to highlight the strengths and weaknesses of each model, providing insights into their suitability for real-world applications. Through detailed comparisons and discussions. As well as, it seeks to identify the most effective deep learning approaches and propose potential improvements for future research.

4.2 Environment and development tools

4.2.1 Interactive computational environments

4.2.1.1 Jupyter notebook

It is an open-source web application in Anaconda but can also be installed independently, it allows users to execute python codes in pieces called ‘cells’ and provide instant results. Additionally, it is well fit for machine and deep learning tasks, equipped with a comprehensive suite of libraries and Python modules, also new libraries can be easily installed only by using ‘pip install’ command in order to build, train, and evaluate models.

The version ‘6.5.4’ is used that offers stable performance for data pre-processing tasks.

4.2.1.2 Google Colab

It is an interactive platform that allows python code execution directly in a web browser with no setup requires and offers free access to GPUs and TPUs for 12 hours, to create and manage notebooks users need a Gmail account. Colab already has important libraries for machine and deep learning and additional libraries can be installed via the "!pip install" command, it supports data and notebook imports from Google Drive.

During the pre-processing phase Jupyter Notebook is used due to its robust performance and integration with Anaconda's environment. However, for implementing and training deep learning models we switch to Google Colab to utilize its GPU capabilities which substantially decreases computation time.

4.2.2 Libraries

4.2.2.1 Libraries for pre-processing

- Numpy[14]: Numpy is short of 'Numerical Python' that is a fundamental package in python that provides logical, shape manipulation, sorting, selecting, discrete Fourier transforms, random simulation and much more.
  Numpy version '1.24.3' is used that handles arrays, performing mathematical computations and managing data types as well as NaN values and converting array types.
- Librosa[15]: It is the key library for audio pre-processing, it is helpful in music or audio analysis.
  Librosa version'0.10.1' is used in order to extract MFCCs features from audio files.
- Soundfile[16]: It is a Python module that can read and write sound files in various formats.
  It is used to make sure that the audio files are properly managed when extracting features.
- Pydub[17]: It allows the application of more complex audio processing effects includes adjusting volumes, and applying audio effects.
- imbalanced-learn[18]: To handle imbalanced dataset, using RandomOverSampler randomly duplicates examples in the minority class.
- Scikit-learn[19]: It is a library for machine learning and deep learning used to split the data into training and testing sets.

4.2.2.2 Libraries for training and testing

- TensorFlow[20]: TensorFlow is an open-source framework developed by Google for high-performance numerical computation, it is used in machine learning and deep learning, capable of running on both CPUs and GPUs.
  version '2.13.0' is used to build and train complex deep learning models
- Keras[21]: It is an API for building and training deep learning models, for now it is fully integrated within TensorFlow as 'tf.keras'. It does simplify many aspects of creating deep learning models through a highly modular and flexible approach.
  Keras models, layers, optimizers and callbacks components are used in order to implement deep learning models.
- Scikit-learn: It is a library for machine learning and deep learning that provides simple and efficient tools for data mining and data analysis.
  It is used to evaluate the model performance using accuracy, precision, f1_score and recall.

[14] https://numpy.org/
[15] librosa — librosa 0.10.2.post1 documentation
[16] https://pypi.org/project/soundfile/
[17] https://pypi.org/project/pydub/
[18] imbalanced-learn documentation — Version 0.12.3
[19] https://scikit-learn.org/stable/index.html
[20] https://www.tensorflow.org/
[21] https://keras.io/

4.2.2.3 Libraries for plotting

- Seaborn[22]: Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- Matplotlib[23]: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
  It is used to plot the training history of each model.

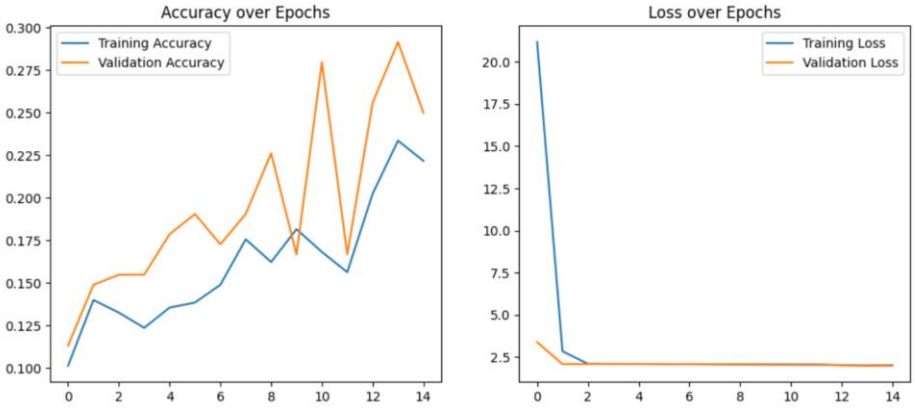
4.3 Results

4.3.1 Trained on row data

The dataset is used directly without segmentation or augmentation (336 audio), using the MFCC and spectrogram combined to extract features, then balancing it (achieving 840 records) then split it into training and testing where Training shape: (672, 1038) and Testing shape: (168, 1038) giving the following results:

4.3.1.1 CNN

The figures bellow shows the evaluation of the CNN model on row data



[22] https://seaborn.pydata.org/
[23] https://matplotlib.org/

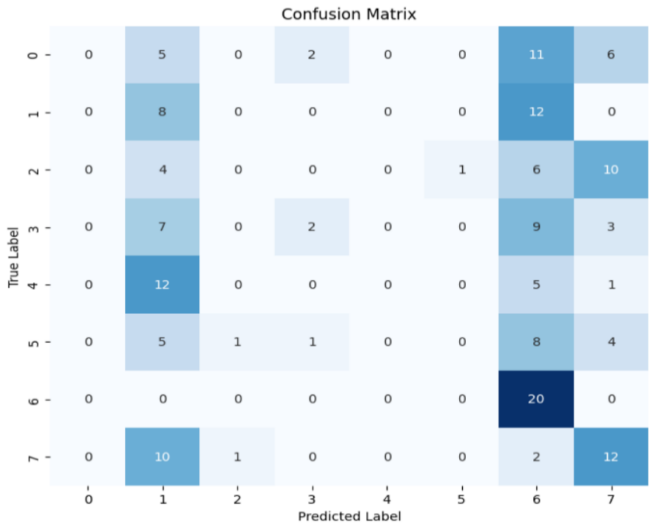


Figure 25: CNN Performance Evaluation on row data

4.3.1.2 LSTM

The figures bellow shows the evaluation of the LSTM model on row data

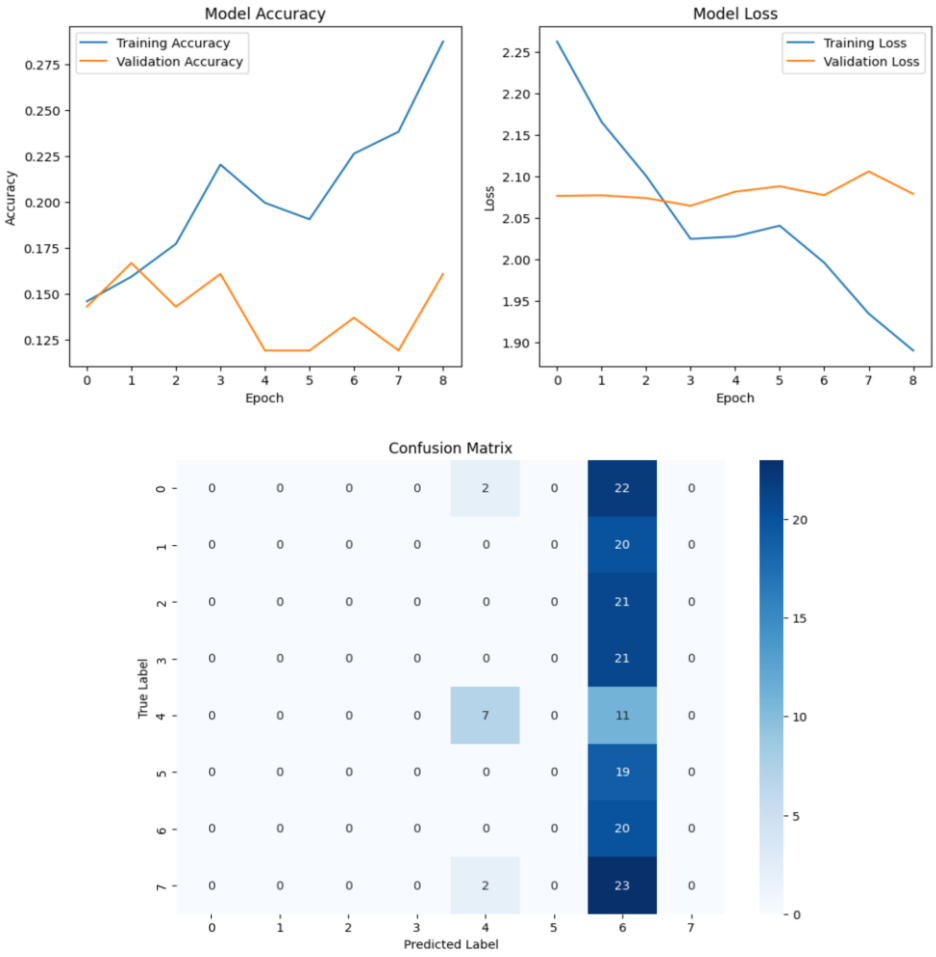


Figure 26: LSTM Performance Evaluation on row data

4.3.1.3 RNN

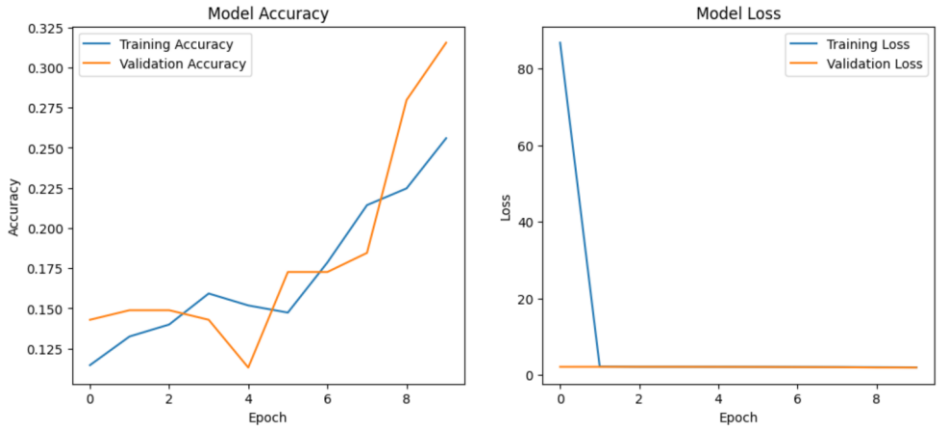
The figures bellow shows the evaluation of the RNN model on row data



Figure 27: RNN Performance Evaluation

4.3.1.4 TDNN

The figures bellow shows the evaluation of the attention model on row data



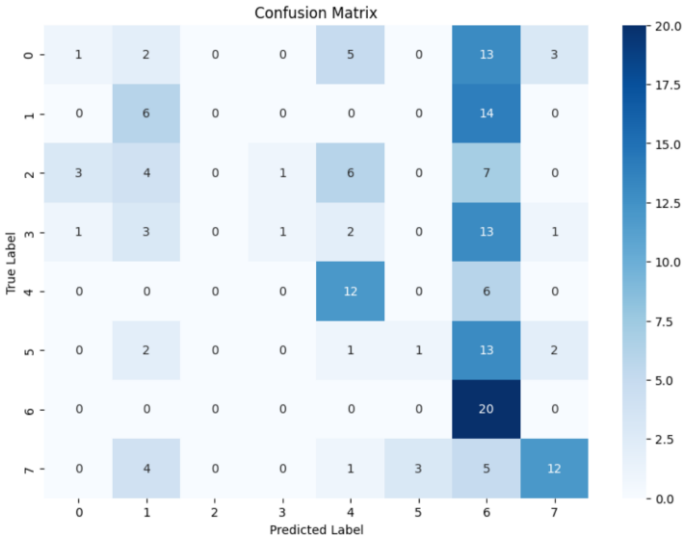


Figure 28: TDNN Performance Evaluation on row data

4.3.1.5 Attention

The figures bellow shows the evaluation of the attention model on row data

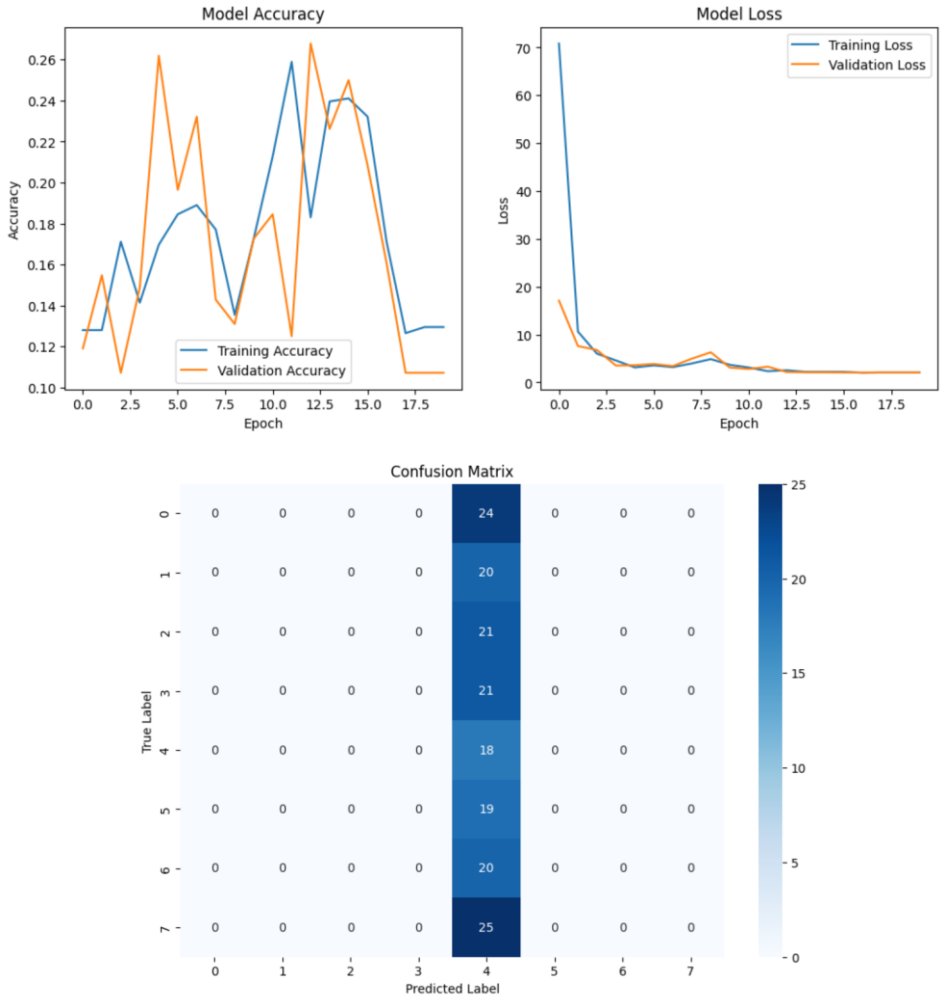


Figure 29: attention Performance Evaluation on row data

4.3.1.6 Autoencoder

The figures bellow shows the evaluation of the autoencoder model on row data

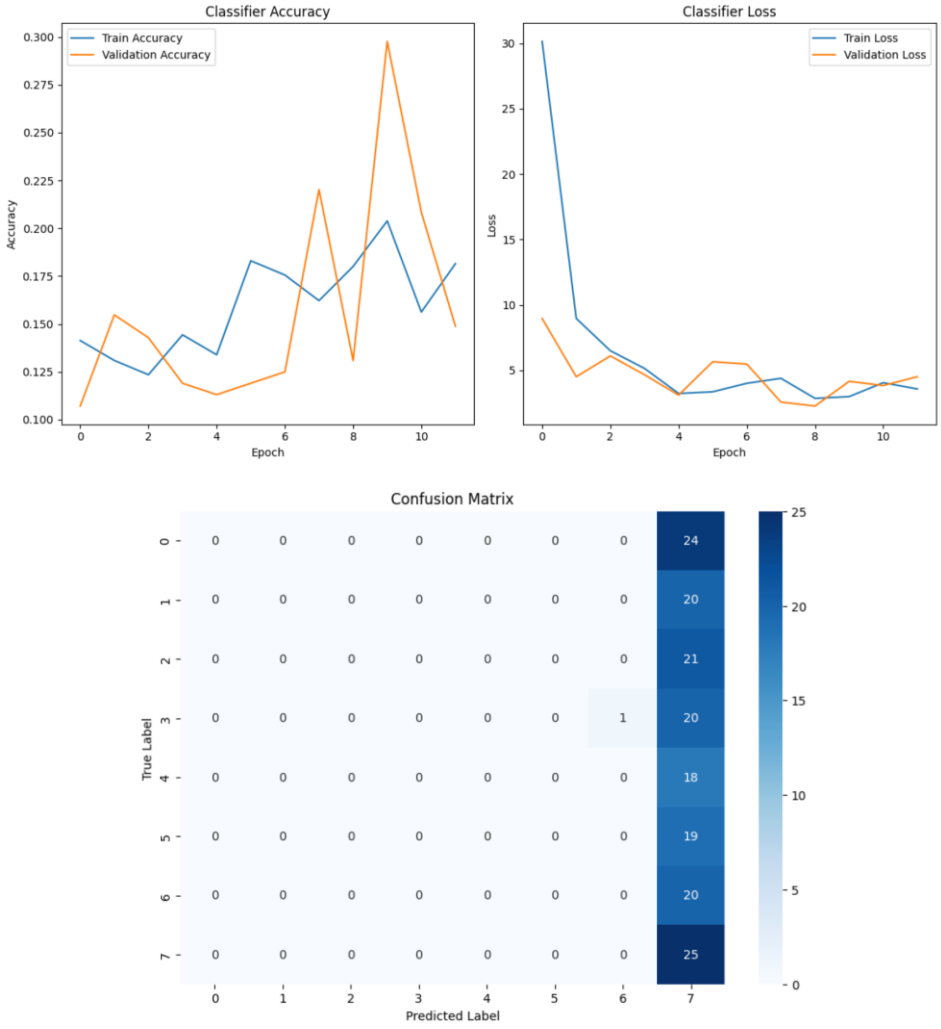
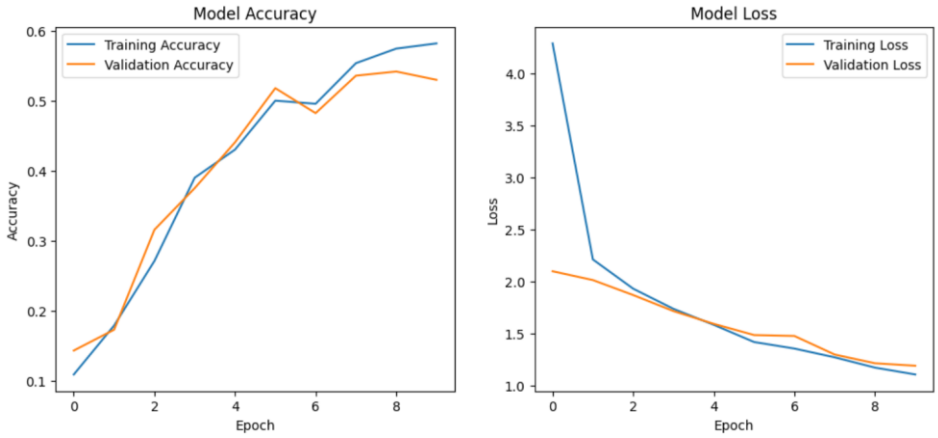


Figure 30: Autoencoder Performance Evaluation on row data

4.3.1.7 Wavenet

The figures bellow shows the evaluation of the Wavenet model on row data



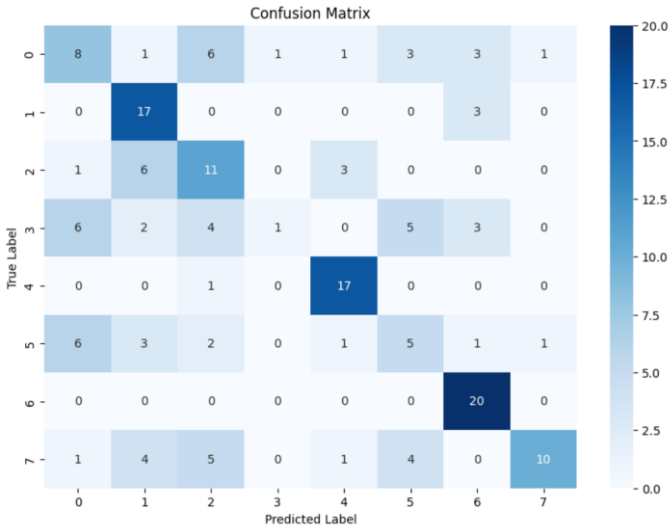


Figure 31: Wavenet Performance Evaluation on row data

the table below show the results:

Models	**Accuracy**	**Precision**	**Recall**	**F1-score**
CNN	25%	15,08%	25%	15,58%
LSTM	16.07%	8.33%	16,08%	7.86%
RNN	41,66%	56,87%	41,66%	35,02%
TDNN	31,54%	32,63%	31,54%	24,84%
Autoencoder	14.88%	2,22%	14.88%	3.87%
Attention	10.71%	1.14%	20.72%	2.73%
wavenet	**52,97%**	**53.90%**	**52.97%**	**48.78%**

Table 9: Evaluation on row data

These models trained on row data that have clearly produced suboptimal results where the best model is Wavenet, there are numerous possibilities for why they are underperforming, we have attempted to resolve these issues in the subsequent phases.

4.3.2 Augmented data

After the disappointing results on row data because of the lack of data (336 audio), augmentation techniques are applied including:

- the combination of voice adjustment and noise the recordings increased to 2016
- the combination of time shifting and dynamic range compression the recordings increased to 336
- Mild pitch shifting the recordings increased to 336
- Equalization the recordings increased to 336

Achieving at the end 3360 audio, then using the combination of both spectrogram and MFCC to extract features. Even after the augmentation the data still imbalanced so Oversampling the Minority Class is implemented by duplicating the minority classes achieving 8400 audio, then split it into training and testing where Training shape:

(6720, 1038) and Testing shape:(1680, 1038), After these pre-processing step the data is ready to be fed to the deep learning models giving the following results:

4.3.2.1 CNN

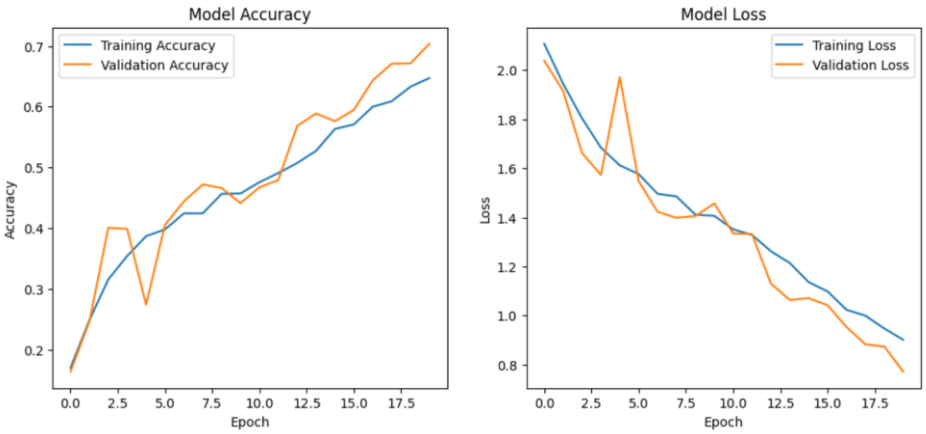
The figures bellow shows the evaluation of the CNN model on augmented data



Figure 32: CNN Performance Evaluation on augmented data

4.3.2.2 LSTM

The figures bellow shows the evaluation of the LSTM model on augmented data



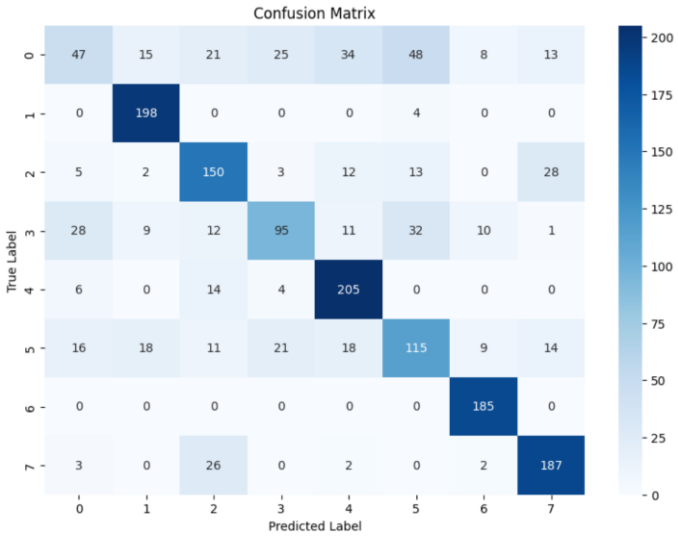
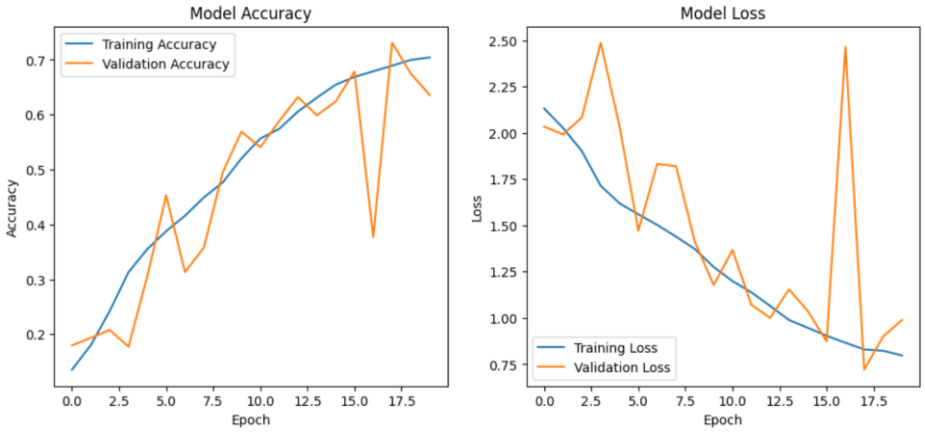


Figure 33: LSTM Performance Evaluation on augmented data

4.3.2.3 RNN

The figures bellow shows the evaluation of the RNN model on augmented data



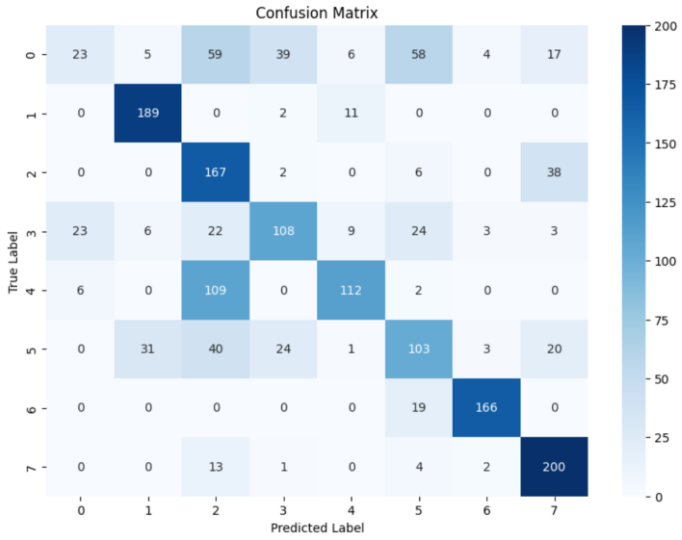


Figure 34: RNN Performance Evaluation on augmented data

4.3.2.4 TDNN

The figures bellow shows the evaluation of the TDNN model on augmented data

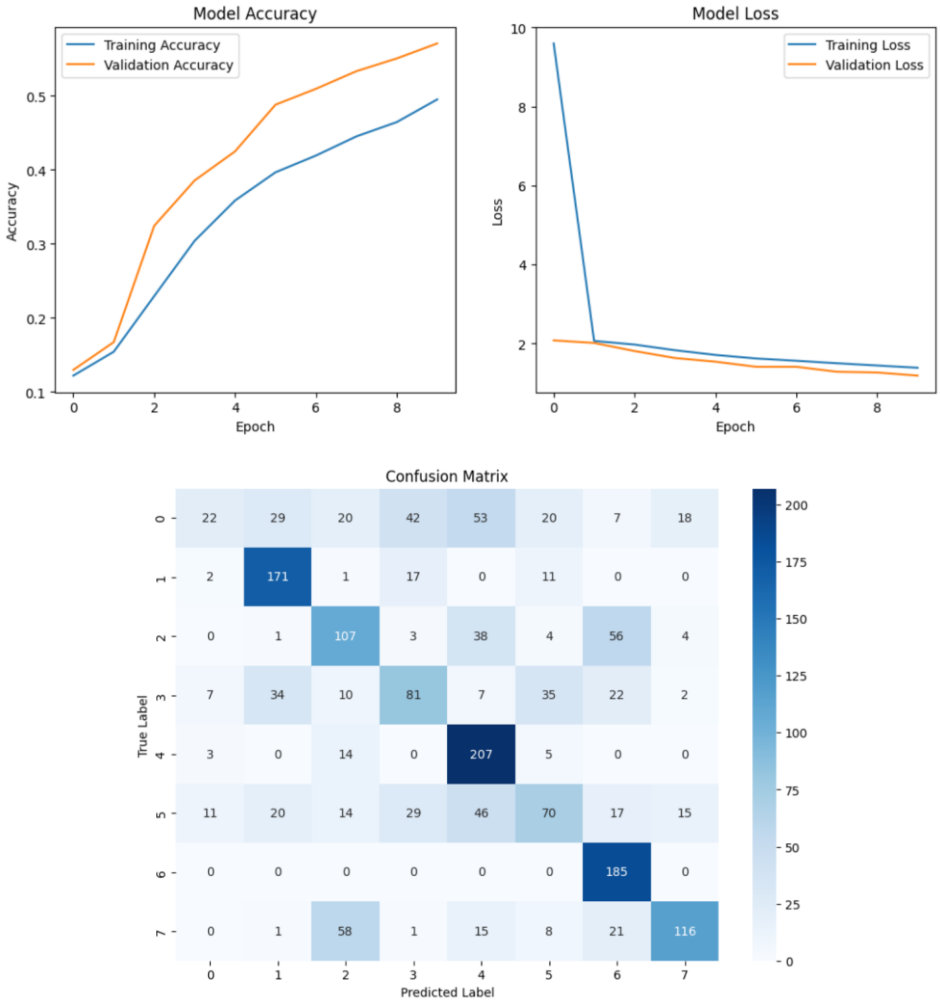


Figure 35: TDNN Performance Evaluation on augmented data

4.3.2.5 Attention

The figures bellow shows the evaluation of the attention model on augmented data

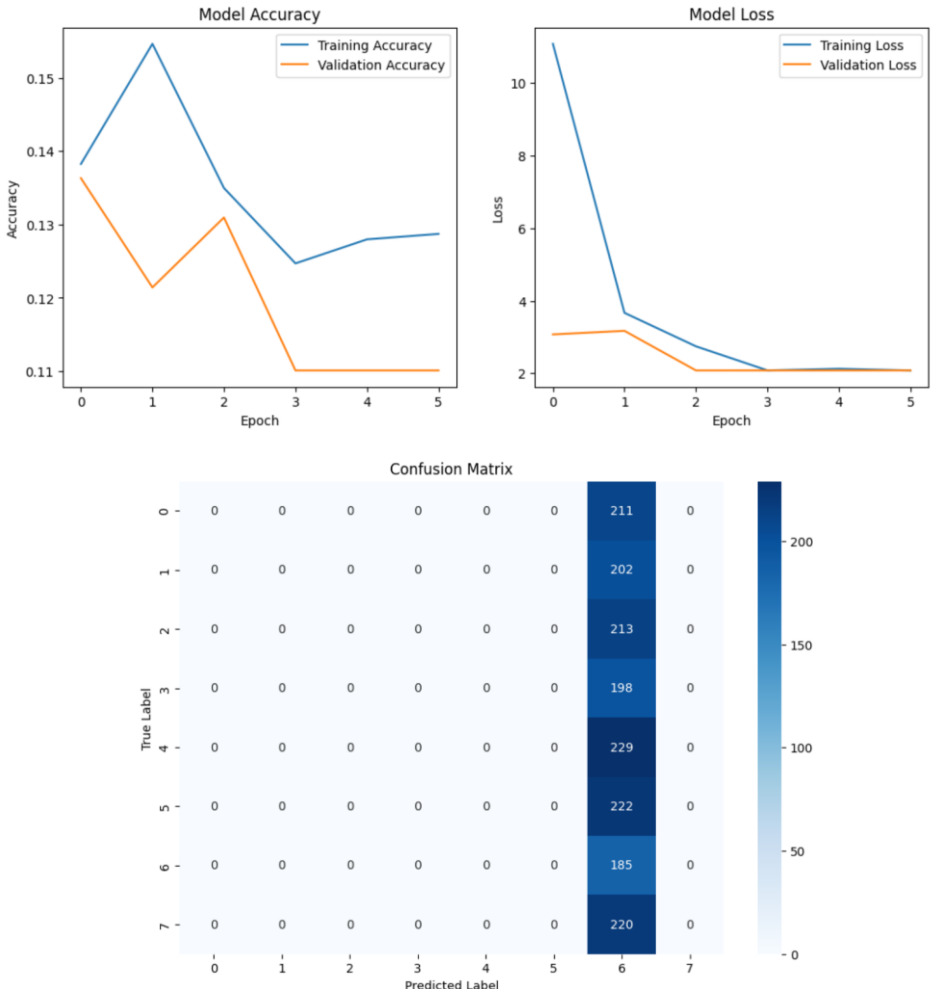
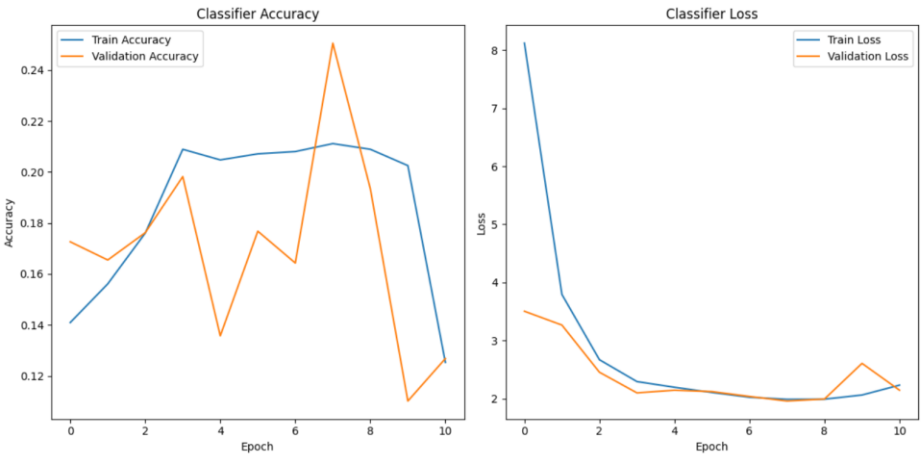


Figure 36: Attention Performance Evaluation on augmented data

4.3.2.6 Autoencoder

The figures bellow shows the evaluation of the autoencoder model on augmented data



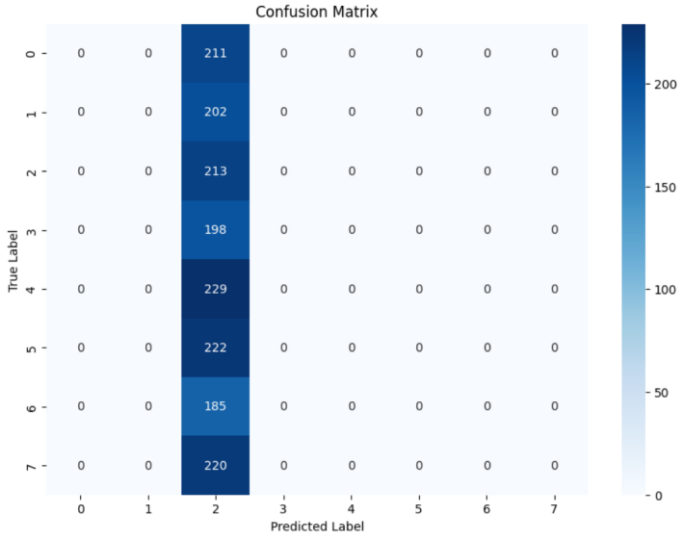


Figure 37: Autoencoder Performance Evaluation on augmented data

4.3.2.7 Wavenet

The figures bellow shows the evaluation of the Wavenet model on augmented data

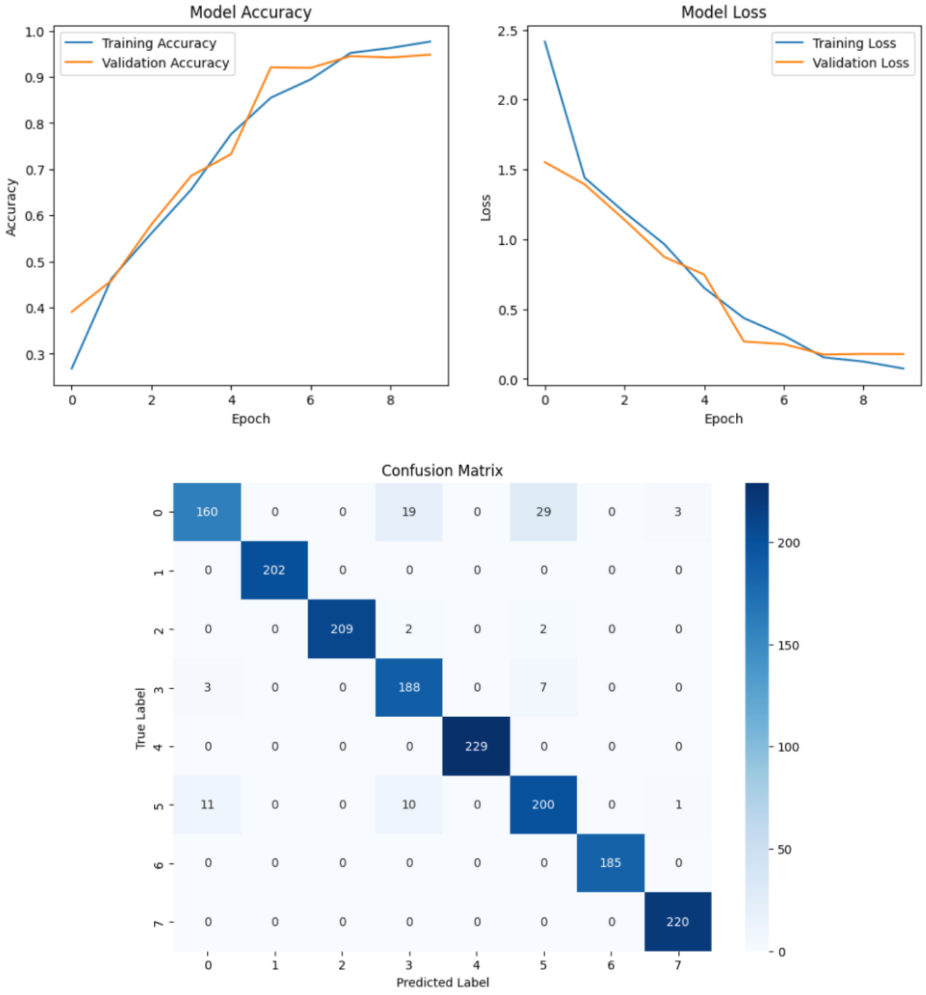


Figure 38: Wavenet Performance Evaluation on augmented data

the table below show the results:

Models	Accuracy	Precision	Recall	F1-score
CNN	63.45%	60.63%	63.45%	59.47%
LSTM	70.35%	67.82%	70.35%	68.17%
RNN	63.57%	64.76%	63.57%	61.39%
TDNN	57.08%	55.91%	57.08%	53.27%
Autoencoder	12.67%	1.60%	12.67%	2.85%
Attention	11.01%	1.21%	11.01%	2.18%
wavenet	94.82%	94.97%	94.82%	94.75%

Table 10: Evaluation on augmented data

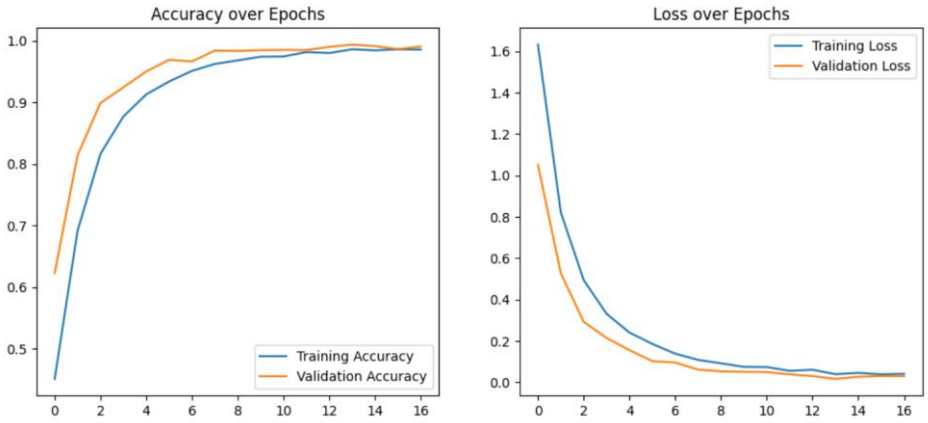
According to the visualisations and the table, there are some models that performs well on the augmented data but wavenet still the best, while there are others that exhibit poor performance such Autoencoder and attention which means that the data still need more pre-processing.

4.3.3 Segmented data

In order to enhance the performance of deep learning models, segmentation is applied to the audios before applying the augmentation techniques, where after segmenting the dataset the number of segmented audios increased to 1604 then after augmentation achieves 14427 recording then using the MFCC and spectrogram combined to extract features, then balancing it and split it into training and testing where Training shape: (19468, 1038) and Testing shape: (4868, 1038) giving the following results:

4.3.3.1 CNN

The figures bellow shows the evaluation of the CNN model on segmented data



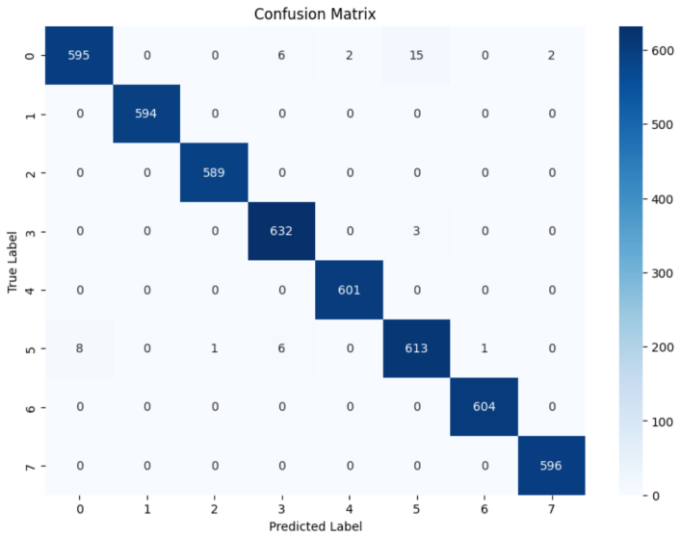


Figure 39:CNN Performance Evaluation on segmented data

4.3.3.2 LSTM

The figures bellow shows the evaluation of the LSTM model on segmented data

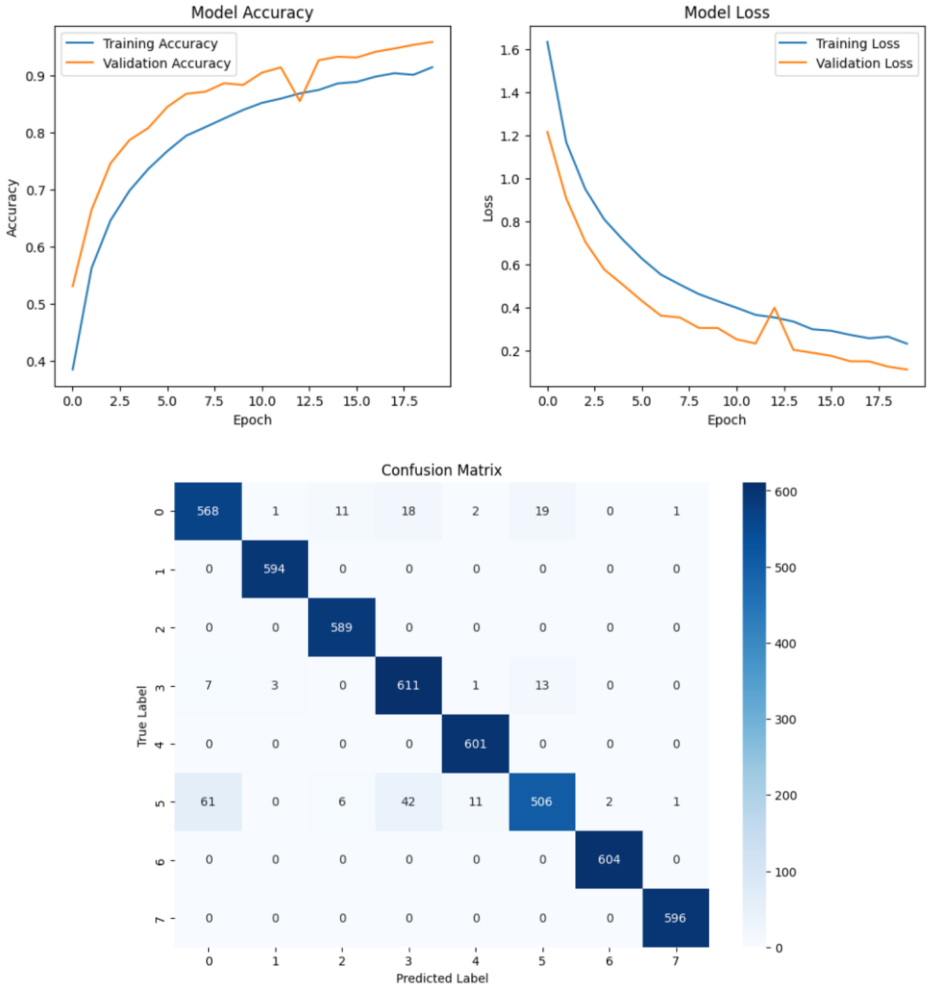


Figure 40: LSTM Performance Evaluation on segmented data

4.3.3.3 RNN

The figures bellow shows the evaluation of the RNN model on segmented data

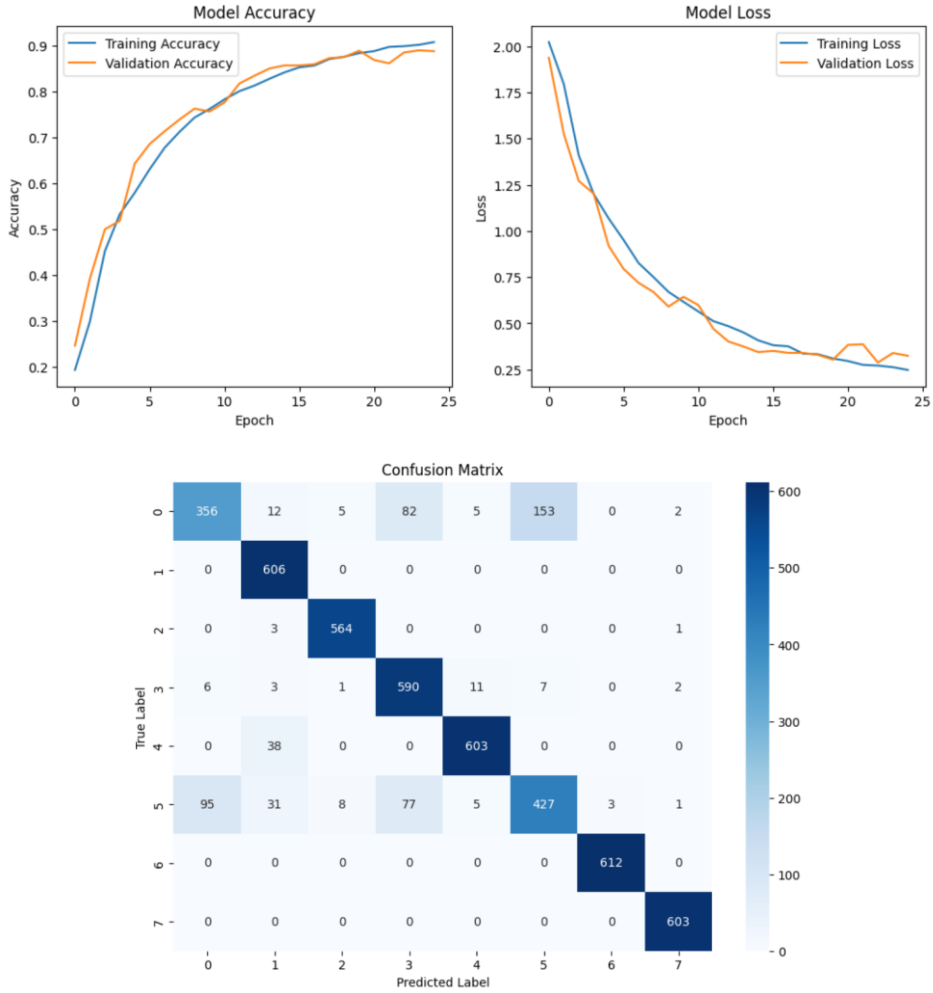
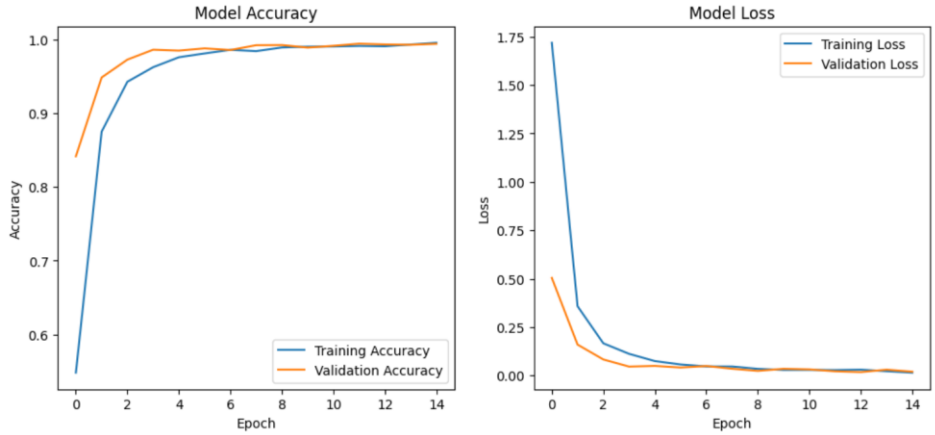


Figure 41: RNN Performance Evaluation on segmented data

4.3.3.4 TDNN

The figures bellow shows the evaluation of the TDNN model



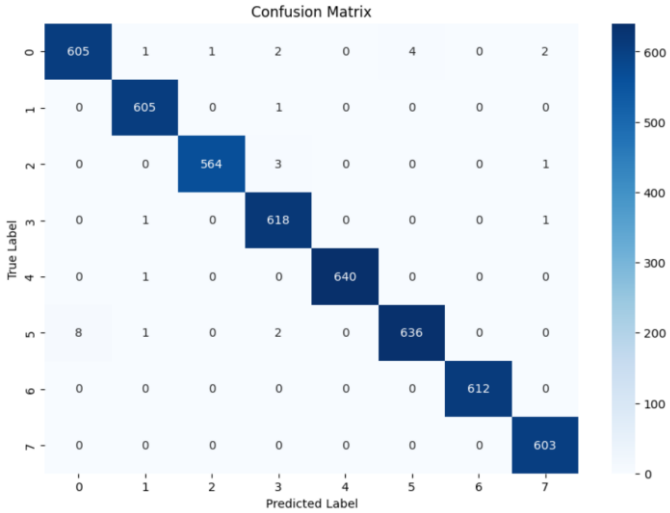


Figure 42: TDNN Performance Evaluation on segmented data

4.3.3.5 Attention

The figures bellow shows the evaluation of the attention model



Figure 43: attention Performance Evaluation on segmented data

4.3.3.6 Autoencoder

The figures bellow shows the evaluation of the autoencoder model

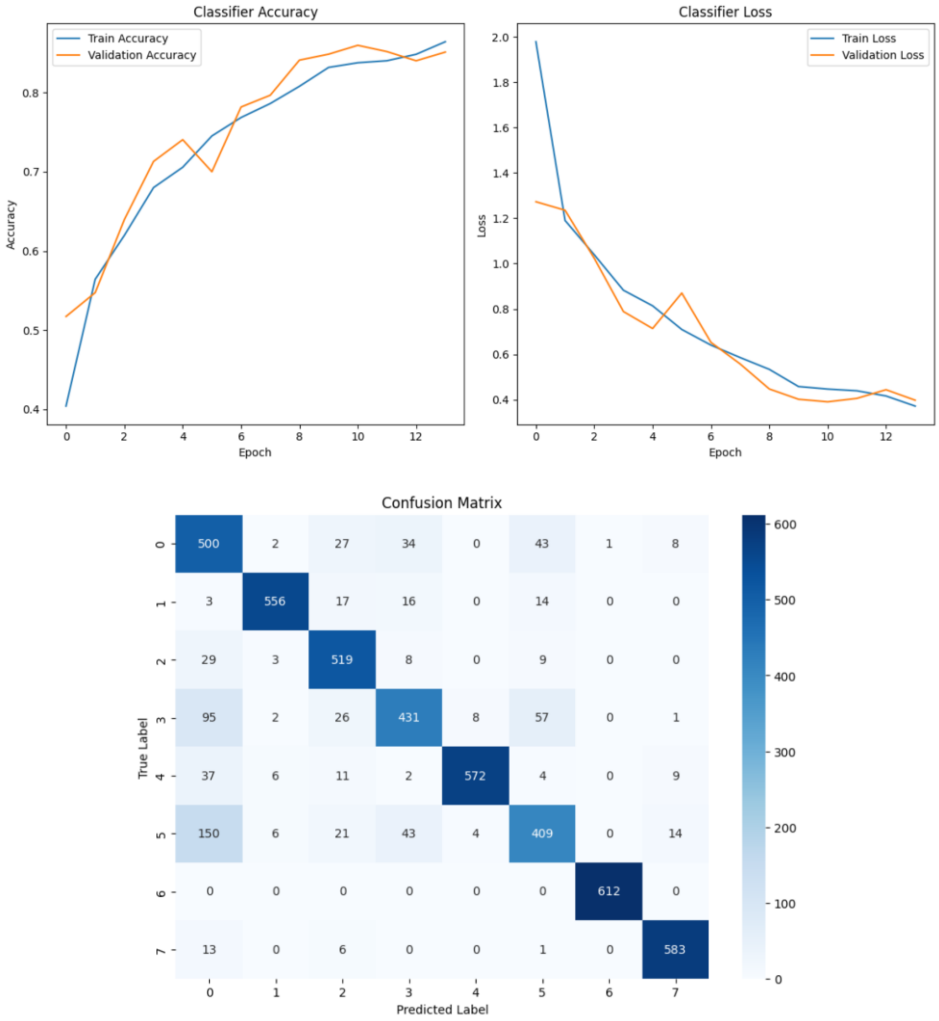
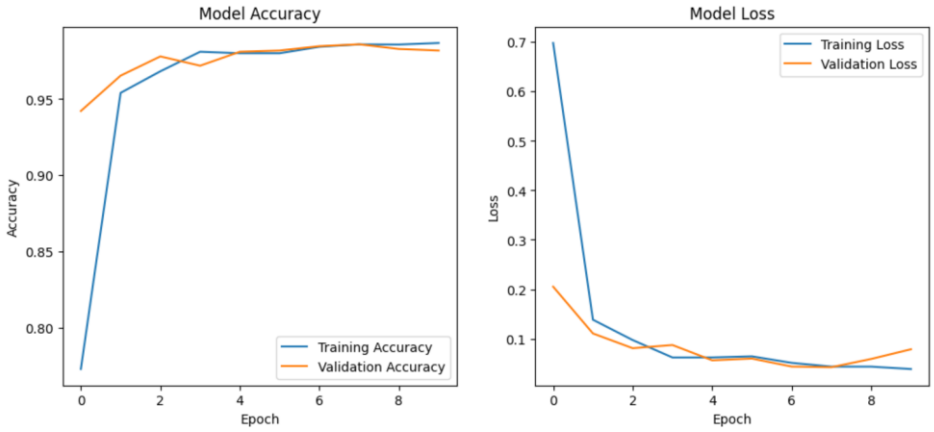


Figure 44:Autoencoder Performance Evaluation on segmented data

4.3.3.7 Wavenet

The figures bellow shows the evaluation of the wavenet model



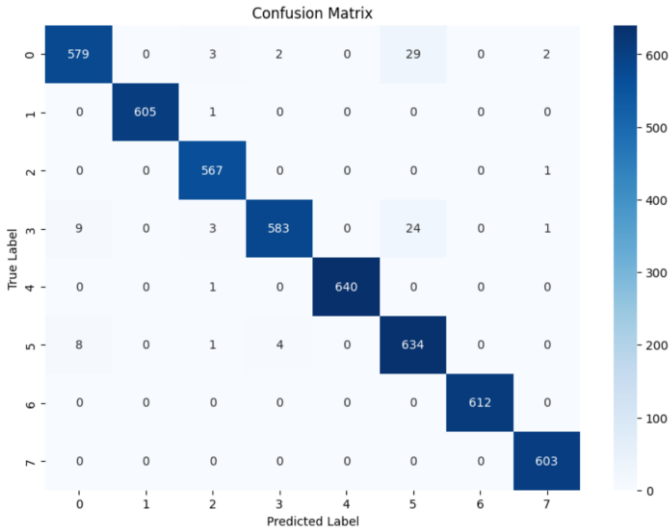
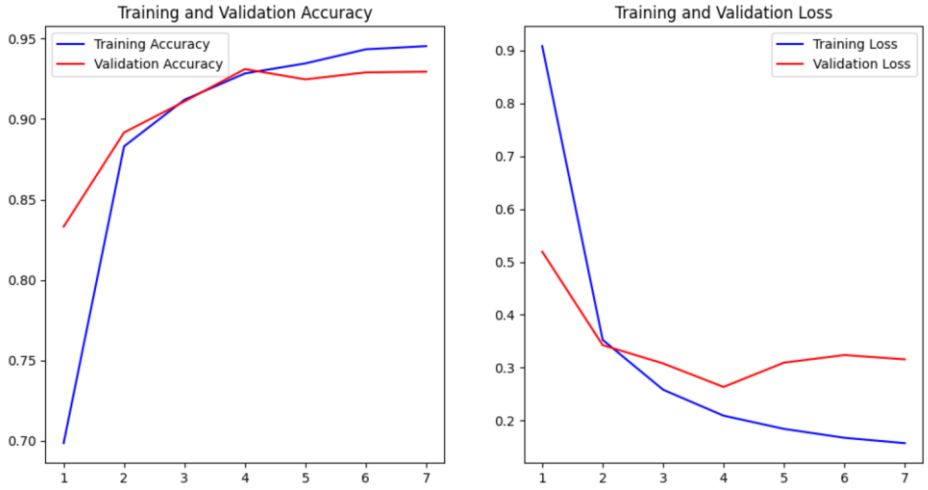


Figure 45: Wavenet Performance Evaluation on segmented data

4.3.3.8 CNN+RNN

The figures bellow shows the evaluation of the hybrid model of CNN and RNN



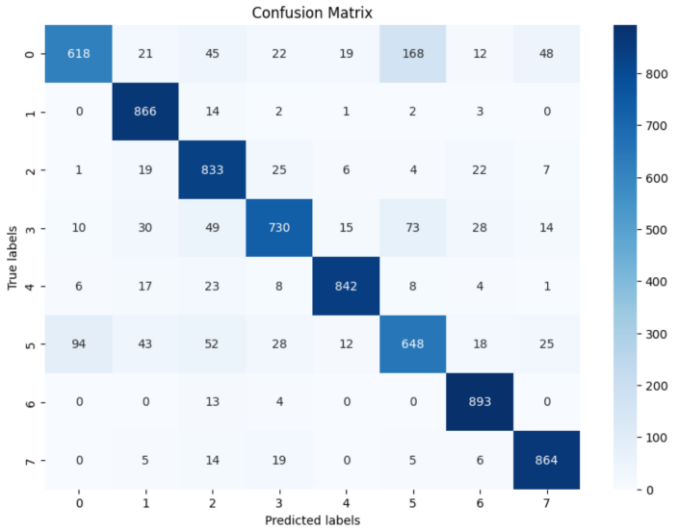


Figure 46: CNN+RNN Performance Evaluation on segmented data

4.3.3.9 CNN+ attention

The figures bellow shows the evaluation of the hybrid model of CNN and attention

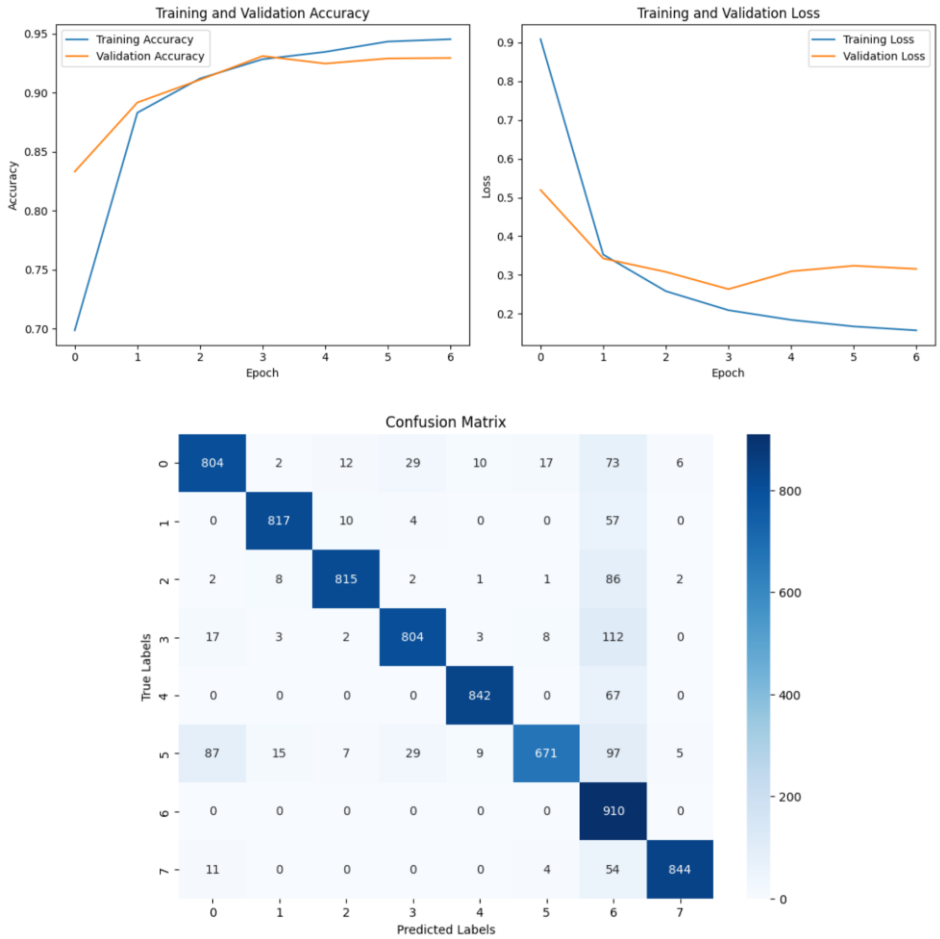


Figure 47: CNN+ Attention Performance Evaluation on segmented data

4.3.3.10 CNN+ Wavenet

The figures bellow shows the evaluation of the hybrid model of CNN and wavenet

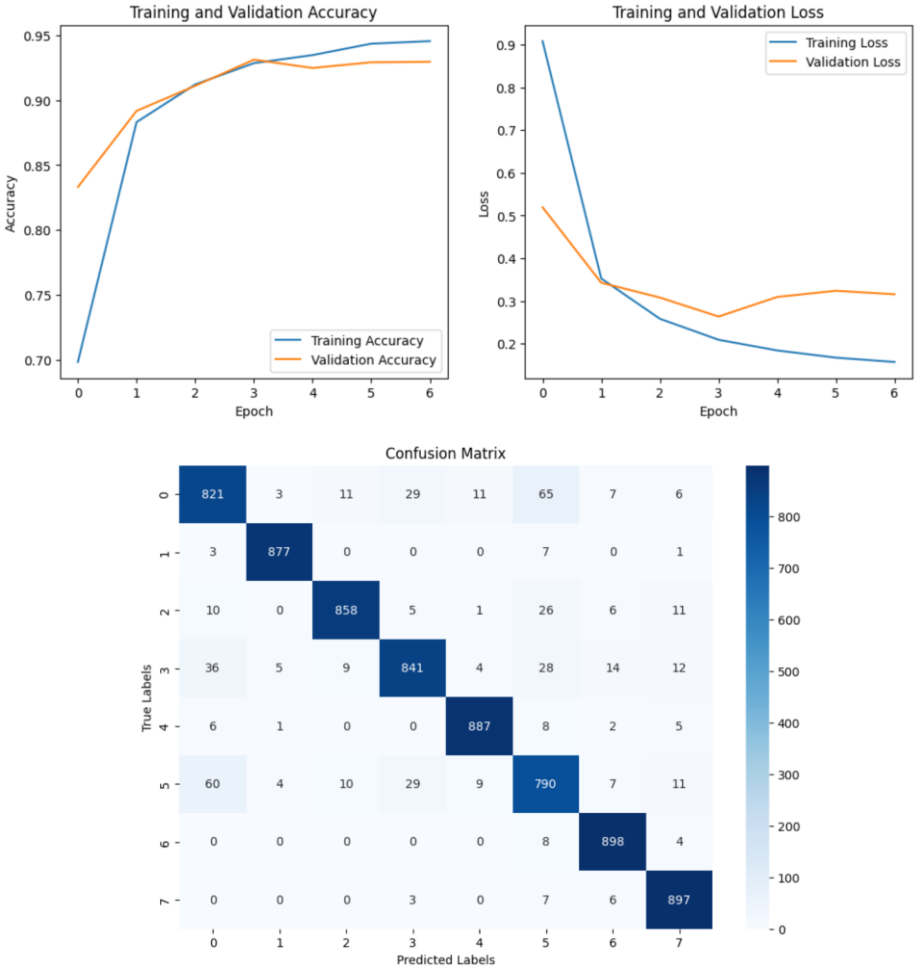
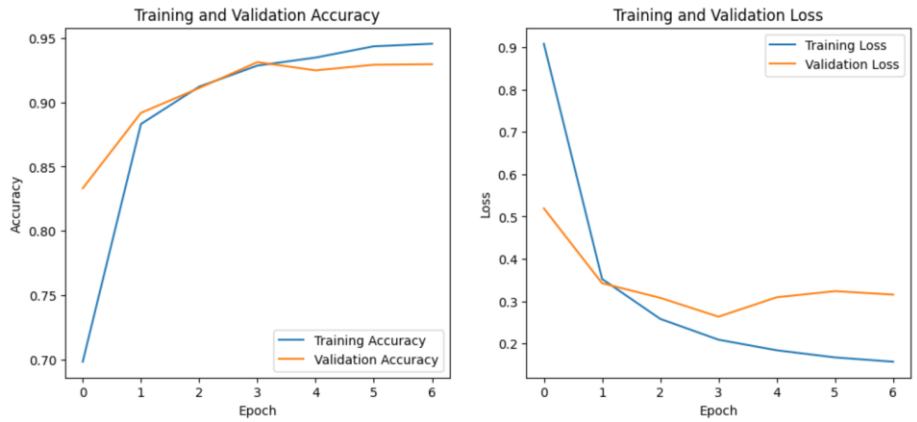


Figure 48: CNN +Wavenet Performance Evaluation on segmented data

4.3.3.11 CNN+ RNN+ attention

The figures bellow shows the evaluation of the hybrid model of CNN and RNN and attention



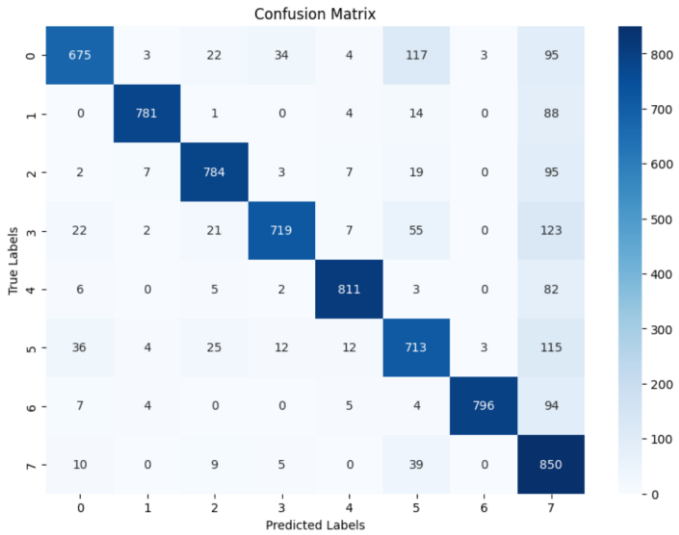


Figure 49: CNN+ RNN +Attention Performance Evaluation on segmented data

the table below show the results:

Models	**Accuracy**	**Precision**	**Recall**	**F1-score**
CNN	99.09%	99.09%	99.09%	99.03%
LSTM	95.91%	95.91%	95.91%	95.83%
RNN	88.78%	88.51%	88.78%	88.28%
TDNN	99.40%	99.41%	99.40%	99.40%
Autoencoder	85.13%	86.14%	85.13%	85.25%
Attention	85.58%	85.36%	85.58%	85.08%
Wavenet	98.18%	98.23	98.18%	98.18%
CNN+ RNN	91.20%	90.98%	91.20%	90.94%
CNN +Attention	97.27%	97.42%	97.27%	97.29%
CNN +Wavenet	98.79%	98.79%	98.78%	98.77%
CNN+ RNN +Attention	90%	90.98%	90%	90%

Table 11: Evaluation on segmented data

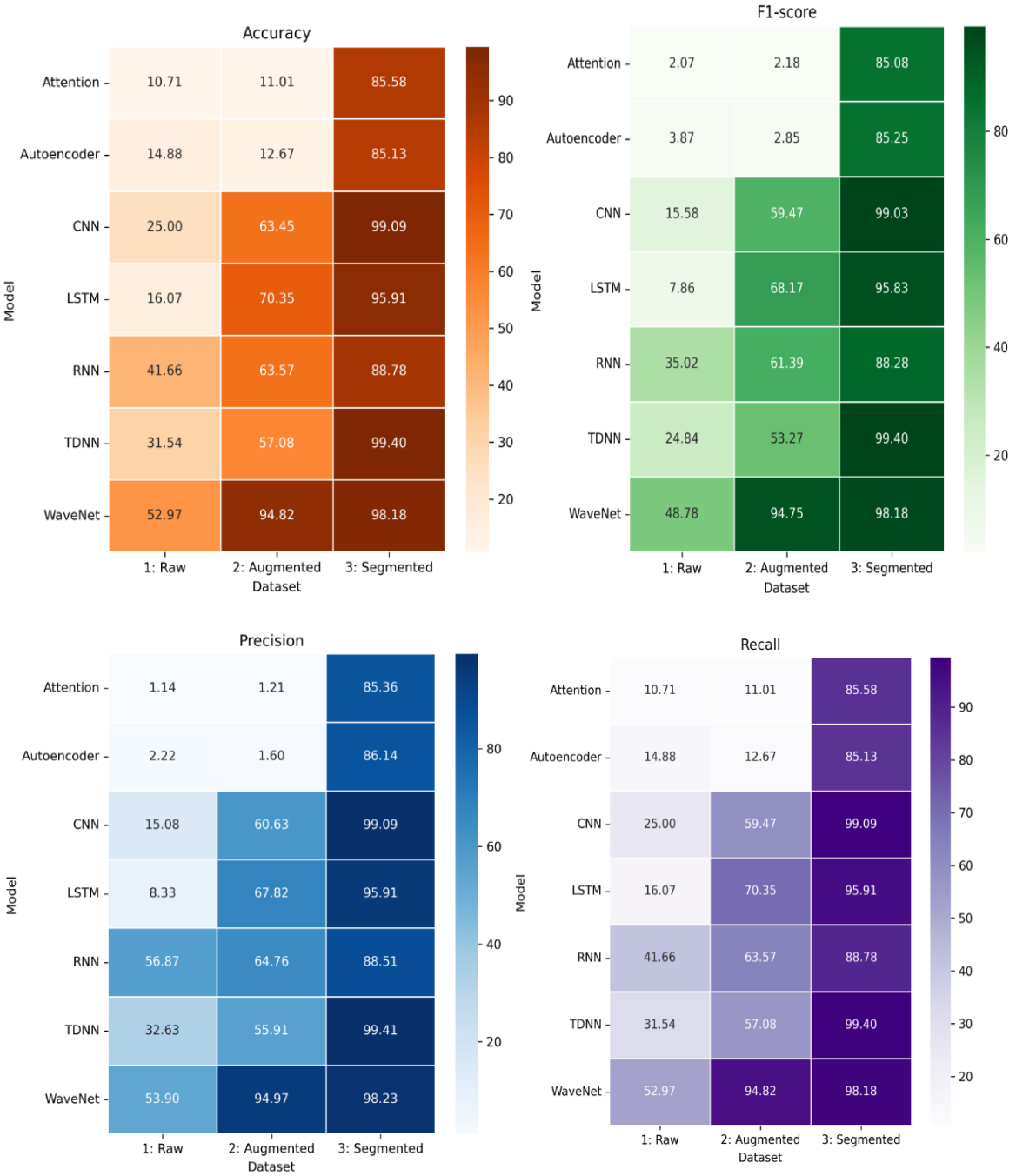


Figure 50: Evaluation on row, augmented and segmented data

		accuracy	*precision*	*recall*	*F1 score*
Articles results	[24]	99.3%	-	-	-
	[25]	90%	-	-	-
	[26]	92.30%	-	-	-
	[27]	93%	-	-	-
	[28]	95%	-	-	-
	[29]	87%	-	86%	-
	[31]	91%	-	86%	-
	[32]	**99.19%**	**-**	**-**	**-**
	[33]	**99.8%**	**-**	**99.8%**	**-**
	[34]	66.31%	-	-	-
Trained on row data	CNN	25%	15,08%	25%	15,58%
	LSTM	16.07%	8.33%	16,08%	7.86%
	RNN	41,66%	56,87%	41,66%	35,02%
	TDNN	31,54%	32,63%	31,54%	24,84%
	Autoencoder	14.88%	2,22%	14.88%	3.87%
	Attention	10.71%	1.14%	20.72%	2.73%
	Wavenet	52,97%	53.90%	52.97%	48.78%
Trained on augmented data	CNN	63.45%	60.63%	63.45%	59.47%
	LSTM	70.35%	67.82%	70.35%	68.17%
	RNN	63.57%	64.76%	63.57%	61.39%
	TDNN	57.08%	55.91%	57.08%	53.27%
	Autoencoder	12.67%	1.60%	12.67%	2.85%
	Attention	11.01%	1.21%	11.01%	2.18%
	Wavenet	94.82%	94.97%	94.82%	94.75%
Trained on segmented data	**CNN**	**99.09%**	**99.09%**	**99.09%**	**99.03%**
	LSTM	95.91%	95.91%	95.91%	95.83%
	RNN	88.78%	88.51%	88.78%	88.28%
	TDNN	**99.40%**	**99.41%**	**99.40%**	**99.40%**
	Autoencoder	85.13%	86.14%	85.13%	85.25%
	Attention	85.58%	85.36%	85.58%	85.08%
	Wavenet	98.18%	98.23	98.18%	98.18%
	CNN+ RNN	91.20%	90.98%	91.20%	90.94%
	CNN +Attention	97.27%	97.42%	97.27%	97.29%
	CNN +Wavenet	98.79%	98.79%	98.78%	98.77%
	CNN+ RNN +Attention	90%	90.98%	90%	90%

Table 12: Comparison between several deep learning models trained on different data

4.4 Discussion

In order to compare the performance of several deep learning models including CNNs, LSTMs, RNNs, TDNNs, Autoencoders, Attention-based models, Wavenet, and combinations of CNNs with RNNs and Attention or Wavenet. The models are evaluated on three types of data: raw, augmented, and segmented, using metrics such as accuracy, precision, recall, and F1 score.

- Row data: The performance of models on raw data is generally the weakest, likely due to the lack of pre-processing which complicates feature extraction. Wavenet excels in handling raw data, thanks to its deep convolutional structure that adeptly manages raw audio signals. RNNs also show decent performance, likely due to their capacity to understand temporal dependencies in unprocessed data streams.
- Augmented data: Most models exhibit improved performance on augmented data compared to raw data. Wavenet continues to excel with augmented data, maintaining high precision and F1 scores, reinforcing its robustness in diverse conditions. Attention-based models also show notable improvement, benefiting from augmented contexts to refine their focus and feature extraction capabilities.
- Segmented data: Almost all models perform best with segmented data. The segmentation likely simplifies the input by focusing on the most relevant parts of the data, thereby enhancing model accuracy and efficiency.

  Models combining CNN with RNN, Attention mechanisms and wavenet show significantly high performance on segmented data. This indicates that the hybrid approach leverages spatial feature extraction of CNNs and the sequential data processing strengths of RNNs, Attention and wavenet making them particularly effective for well-structured and segmented inputs.

  The differences in model performance across the data types underscore the importance of appropriate data pre-processing. Segmenting data, in particular, appears most effective where:

  Wavenet's consistently high performance across all data types highlights its strength in dealing with audio data, benefiting from its ability to model temporal sequences effectively.

  The superior performance of hybrid models on segmented data suggests that combining multiple neural network architectures can capitalize on their individual strengths, offering a powerful approach for complex data analysis tasks.

  Both The standout models are CNN, TDNN, Wavenet, and the combined CNN with Wavenet, each achieving over 98% in accuracy, precision, F1 score, and recall,[35].

4.5 Conclusion

This chapter highlights the varying performance of different deep learning models and hybrid models on raw, augmented, and segmented data for respiratory disorder detection. The results show that data segmentation significantly improves model accuracy, while augmentation benefits some models more than others, the best models are TDNN,CNN.

General conclusion

General conclusion

This study represents a significant contribution to the intersection of healthcare and artificial intelligence, focusing on the early detection of various respiratory disorders. By adeptly utilizing sophisticated deep learning models such as CNN, LSTM, RNN, TDNN, Wavenet, and various hybrid combinations like RNN with CNN, and CNN with Wavenet, applied to audio recordings, this research has demonstrated the profound potential of AI in medical diagnostics.

The findings underscore the critical role of proper data pre-processing, with segmentation proving particularly beneficial. Notably, Wavenet consistently outperformed other models across all data types, underscoring its effectiveness in processing complex audio data. Likewise, hybrid models showed superior performance on segmented data, illustrating the advantages of integrating multiple neural architectures for tackling complex data analysis tasks in healthcare. Standout models, including CNN, TDNN, Wavenet, and combinations like CNN with Wavenet, achieved impressive accuracy metrics exceeding 98%.

Looking ahead, we plan to significantly expand the scope and impact of this research by exploring a broader range of models across more diverse datasets and diseases. Future efforts will not only focus on detecting the presence of diseases but also on assessing their severity. This progression is expected to yield crucial insights for clinical treatment and management, refining the accuracy and utility of these models in real-world scenarios. By extending the range of diseases and datasets studied, the aim is to develop more personalized and effective healthcare solutions. This endeavour will involve rigorous testing and adaptation of the models to ensure their robustness and reliability under various conditions, thereby enhancing their applicability in clinical settings and potentially improving patient outcomes globally.

References

[1] V. S. Karkhanis and J. M. Joshi, “Pleural effusion: Diagnosis, treatment, and management,” *Open Access Emergency Medicine*, vol. 4, pp. 31–52, May 2012, doi: 10.2147/OAEM.S29942.

[2] S. Aggarwal *et al.*, “Audio Segmentation Techniques and Applications Based on Deep Learning,” *Sci Program*, vol. 2022, p. 7994191, 2022, doi: 10.1155/2022/7994191.

[3] “INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY.” [Online]. Available: www.ijaret.org

[4] Z. Kh. Abdul and A. K. Al-Talabani, “Mel Frequency Cepstral Coefficient and its Applications: A Review,” *IEEE Access*, vol. 10, pp. 122136–122158, 2022, doi: 10.1109/ACCESS.2022.3223444.

[5] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. LeCun, “Very deep multilingual convolutional neural networks for LVCSR,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4955–4959. doi: 10.1109/ICASSP.2016.7472620.

[6] T. Gorach, “DEEP CONVOLUTIONAL NEURAL NETWORKS-A REVIEW,” *International Research Journal of Engineering and Technology*, 2008, [Online]. Available: www.irjet.net

[7] W. You, C. Shen, D. Wang, L. Chen, X. Jiang, and Z. Zhu, “An Intelligent Deep Feature Learning Method With Improved Activation Functions for Machine Fault Diagnosis,” *IEEE Access*, vol. PP, p. 1, Jun. 2019, doi: 10.1109/ACCESS.2019.2962734.

[8] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[9] A. Graves, “Long Short-Term Memory,” in *Supervised Sequence Labelling with Recurrent Neural Networks*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 37–45. doi: 10.1007/978-3-642-24797-2_4.

[10] C. B. Vennerød, A. Kjærran, and E. S. Bugge, “Long Short-term Memory RNN,” May 2021, [Online]. Available: http://arxiv.org/abs/2105.06756

[11] P. J. Werbos, “Applications of advances in nonlinear sensitivity analysis,” in *System Modeling and Optimization*, F. Drenick R. F. and Kozin, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1982, pp. 762–770.

[12] “Recurrent Neural Network and its Various Architecture Types Trupti Katte,” 2018. [Online]. Available: www.rsisinternational.org

[13] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, “Phoneme recognition using time-delay neural networks,” *IEEE Trans Acoust*, vol. 37, no. 3, pp. 328–339, 1989, doi: 10.1109/29.21701.

[14] B. Liu, W. Zhang, X. Xu, and D. Chen, “Time Delay Recurrent Neural Network for Speech Recognition,” in *Journal of Physics: Conference Series*, Institute of Physics Publishing, May 2019. doi: 10.1088/1742-6596/1229/1/012078.

[15] P. Sharma, K. Sarma, and N. Mastorakis, “Artificial Intelligence Aided Electronic Warfare Systems-Recent Trends and Evolving Applications,” *IEEE Access*, vol. 8, p. 1, Jun. 2020, doi: 10.1109/ACCESS.2020.3044453.

[16] A. Vaswani *et al.*, “Attention Is All You Need,” Jun. 2017, [Online]. Available: http://arxiv.org/abs/1706.03762

[17] Z. Niu, G. Zhong, and H. Yu, “A review on the attention mechanism of deep learning,” *Neurocomputing*, vol. 452, pp. 48–62, 2021, doi: https://doi.org/10.1016/j.neucom.2021.03.091.

[18] C. Liu, H. Raza, and S. Bhattacharyya, “9 - Deep learning methods for analysis of neural signals: From conventional neural network to graph neural network,” in *Advanced Methods in Biomedical Signal Processing and Analysis*, K. Pal, S. Ari, A. Bit, and S. Bhattacharyya, Eds., Academic Press, 2023, pp. 205–242. doi: https://doi.org/10.1016/B978-0-323-85955-4.00010-7.

[19] D. P. Kingma and M. Welling, “Auto-Encoding Variational Bayes,” Dec. 2013, [Online]. Available: http://arxiv.org/abs/1312.6114

[20] N. Patel, S. Patel, and S. H. Mankad, “Impact of autoencoder based compact representation on emotion detection from audio,” *J Ambient Intell Humaniz Comput*, vol. 13, no. 2, pp. 867–885, Feb. 2022, doi: 10.1007/s12652-021-02979-3.

[21] N. Patel, S. Patel, and S. H. Mankad, “Impact of autoencoder based compact representation on emotion detection from audio,” *J Ambient Intell Humaniz Comput*, vol. 13, no. 2, pp. 867–885, Feb. 2022, doi: 10.1007/s12652-021-02979-3.

[22] H. Wang, F. Gao, Y. Zhao, and L. Wu, “WaveNet With Cross-Attention for Audiovisual Speech Recognition,” *IEEE Access*, vol. 8, pp. 169160–169168, 2020, doi: 10.1109/ACCESS.2020.3024218.

[23] A. C. Mazari, N. Boudoukhani, and A. Djeffal, “BERT-based ensemble learning for multi-aspect hate speech detection,” *Cluster Comput*, vol. 27, no. 1, pp. 325–339, 2024, doi: 10.1007/s10586-022-03956-x.

[24] N. S. Haider and A. K. Behera, “Computerized lung sound based classification of asthma and chronic obstructive pulmonary disease (COPD),” *Biocybern Biomed Eng*, vol. 42, no. 1, pp. 42–59, 2022, doi: https://doi.org/10.1016/j.bbe.2021.12.004.

[25] “pmc_8116184”.

[26] T. Albiges, Z. Sabeur, and B. Arbab-Zavar, “Compressed Sensing Data with Performing Audio Signal Reconstruction for the Intelligent Classification of Chronic Respiratory Diseases,” *Sensors*, vol. 23, no. 3, 2023, doi: 10.3390/s23031439.

[27] A. Srivastava, S. Jain, R. Miranda, S. Patil, S. Pandya, and K. Kotecha, “Deep learning based respiratory sound analysis for detection of chronic obstructive pulmonary disease,” *PeerJ Comput Sci*, vol. 7, p. e369, Feb. 2021, doi: 10.7717/peerj-cs.369.

[28] R. Zulfiqar, F. Majeed, R. Irfan, H. T. Rauf, E. Benkhelifa, and A. N. Belkacem, “Abnormal Respiratory Sounds Classification Using Deep CNN Through Artificial Noise Addition,” *Front Med (Lausanne)*, vol. 8, Nov. 2021, doi: 10.3389/fmed.2021.714811.

[29] T. Aptekarev, V. Sokolovsky, E. Furman, N. Kalinina, and G. Furman, “Application of deep learning for bronchial asthma diagnostics using respiratory sound recordings,” *PeerJ Comput Sci*, vol. 9, p. e1173, Jan. 2023, doi: 10.7717/peerj-cs.1173.

[30] S. Gairola, F. Tom, N. Kwatra, and M. Jain, “RespireNet: A Deep Neural Network for Accurately Detecting Abnormal Lung Sounds in Limited Data Setting,” in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2021, pp. 527–530. doi: 10.1109/EMBC46164.2021.9630091.

[31] K. S. Alqudaihi *et al.*, “Cough Sound Detection and Diagnosis Using Artificial Intelligence Techniques: Challenges and Opportunities,” *IEEE Access*, vol. 9, pp. 102327–102344, 2021, doi: 10.1109/ACCESS.2021.3097559.

[32] I. Topaloglu *et al.*, “Explainable attention ResNet18-based model for asthma detection using stethoscope lung sounds,” *Eng Appl Artif Intell*, vol. 126, p. 106887, 2023, doi: https://doi.org/10.1016/j.engappai.2023.106887.

[33] O. Karaarslan, K. D. Belcastro, and O. Ergen, “Respiratory sound-base disease classification and characterization with deep/machine learning techniques,” *Biomed Signal Process Control*, vol. 87, p. 105570, 2024, doi: https://doi.org/10.1016/j.bspc.2023.105570.

[34] J. Acharya and A. Basu, “Deep Neural Network for Respiratory Sound Classification in Wearable Devices Enabled by Patient Specific Model Tuning,” *IEEE Trans Biomed Circuits Syst*, vol. 14, no. 3, pp. 535–544, 2020, doi: 10.1109/TBCAS.2020.2981172.

[35] H. Kheddar, A. C. Mazari, and G. H. Ilk, “Speech Steganography based on Double Approximation of LSFs Parameters in AMR Coding,” in *2022 7th International Conference on Image and Signal Processing and their Applications (ISPA)*, 2022, pp. 1–8. doi: 10.1109/ISPA54004.2022.9786313.

The medical information were taken from World health organization (WHO) World Health Organization (WHO) on June 2024