50.039 Theory and Practice of Deep Learning

Final Project Report

Group 20 - Cough Audio Analysis for COVID-19 Detection

Lee Chang Zheng, Lee Cheng Xin, Jason Peng Jing Ming

Github Repository

Table of Contents

1. Introduction

The COVID-19 pandemic has spurred the exploration of innovative diagnostic methods to combat the spread of the virus. Traditional approaches to diagnosing COVID-19, such as PCR testing, while reliable, often require specialised equipment and a trained personnel to perform the test. Moreover, these tests are time consuming, physically invasive, and may not be readily accessible in all regions. Therefore, alternative solutions for diagnostic methods for COVID-19 are imperative in the fight against the spread of the virus. Among these solutions, analysing cough audio samples emerges as a promising method. Coughing is a common symptom of respiratory illnesses, including COVID-19, and it might aid in early detection of the virus.

Before embarking on the project, we conducted a literature review on using audio data to train neural networks. We found that one of the main methods for training neural networks on audio data involves converting audio waveforms into spectrograms and feeding them into Convolutional Neural Networks (CNNs). Spectrograms are a visual representation of the frequency and amplitude of the audio signals over time. They are essentially images representing the audio data, enabling the CNN to learn features from an audio file. This method has been widely adopted in various audio-related tasks, such as automatic speaker recognition, showcasing that the CNN has the capability to learn and make accurate classification through this method.

This project aims to create a CNN capable of detecting whether a person is healthy or is positive for COVID-19 based on cough audio samples. To achieve this task, we first standardised the audio samples to the same format during data preprocessing. Subsequently, we converted the audio samples into spectrograms containing an image representation of the audio data. Finally, we pass these spectrograms into the CNN model for training and classification.

2. Instructions

Before starting, clone the GitHub repository into your desired folder with the following command:

```
git clone https://github.com/Leecz888/50.039-DL-Project.git
```

This is the folder structure of our code:

```
.
├── 📁 Inputs/                  # Contains the audio files from the dataset
│   ├── 🔉 audio_file.ogg
│   ├── 🔉 audio_file.webm
│   ├── …
├── 📁 Outputs/                 # Contains the pre-processed audio files
│   ├── 🔉 audio_file.wav
│   ├── …
├── 📒 model.ipynb              # Main code
├── 🐍 noise_reduction.py       # Contains code to perform noise reduction on the audio files
├── 🐍 silence.py               # Contains code to remove silence from the audio files
├── 🎨 Deep learning.drawio     # Contains the system architecture diagram
├── ⚙ .gitignore
├── 📄 Dataset.xlsx             # Contains the metadata information for the dataset
├── 📄 requirements.txt
└── 📄 README.md
```

The “Inputs” folder contains all the audio clips from the original dataset. The “Outputs” folder will be used to store the audio clips after the audio preprocessing process in model.ipynb. model.ipynb contains all the necessary code for the project. The required dependencies for our project are located under requirements.txt.

2.1 Setting up a virtual environment

Before running the code in model.ipynb, it is recommended to set up a virtual environment to install the required dependencies for this project. Please install Conda in your system if it is not already installed. Follow the instructions in the official Conda documentation for installing Conda for your operating system. While we used Conda to set up the virtual environment, there are other ways of setting up a

virtual environment as well, however the following instructions for creating a virtual environment are applicable only for Conda.

a) Creating a virtual environment in Conda

To create a virtual environment in Conda, run the following command:

conda create -n <ENVIRONMENT NAME> python

Replace <ENVIRONMENT NAME> with your desired name for the virtual environment. Follow the instructions in the command line interface to finish creating the virtual environment

b) Activating the virtual environment

Upon creating the virtual environment, we will need to activate it with the following command:

conda activate <ENVIRONMENT NAME>

Where <ENVIRONMENT NAME> is the name of the virtual environment you created.

c) Deactivating the virtual environment

To deactivate your virtual environment, run the following command:

conda deactivate <ENVIRONMENT NAME>

Where <ENVIRONMENT NAME> is the name of the virtual environment you created.

d) Deleting the virtual environment

To delete your virtual environment, run the following command:

conda remove -n <ENVIRONMENT NAME> --all

Where <ENVIRONMENT NAME> is the name of the virtual environment you created. This will delete the entire virtual environment and all the dependencies installed in it.

2.2 Installing the required dependencies

Upon activating your virtual environment, we can then install the required dependencies located under requirements.txt. Run the following command in your virtual environment:

pip install -r requirements.txt

This will install all the required dependencies for this project.

Apart from this, it might be good to install FFmpeg before running the project. As our project works with audio files, FFmpeg is a required dependency in most audio related tasks such as audio conversion. While

we have already converted the audio files to the correct format, FFmpeg would be required if you wish to run the cell for the audio conversion.

Installing FFmpeg is rather lengthy depending on your operating system. As such, we recommend following a guide if you are not sure. The link to the official FFmpeg download page can be found here.

After installing all the required dependencies, you are now ready to run the code!

3. Dataset

We used the COUGHVID dataset for the purposes of this project. COUGHVID is a crowdsourced dataset and the corpus was created for the study of large-scale cough analysis algorithms. The dataset contains over 20,000 crowdsourced cough recordings representing a wide range of ages, genders, geographic locations, and COVID-19 statuses. The link to the publicly available dataset can be found here.

3.1 Problems with the Dataset

For the project, we only used a subset of the dataset. In total, we only used 1,645 files out of the 20,000 audio clips. This is because of 4 key reasons:

1) Unlabelled data

   While examining the data, we realised that a large chunk of the dataset has not been properly labelled. As our task involves classifying individuals with and without COVID-19, we were most interested in the "status" column, which indicates whether a person has COVID-19 or is healthy. However, under the "status" column, only approximately 10,000 audio files have a labelled status. As seen in Figure 1, row 11316 onwards contains an empty field in the status column.

1	uuid	datetime	cough_detected	latitude	longitude	age	gender	respiratory_condition	fever_muscle_pain	status
11310	ff201807-	2020-04-1	0.4831			55	male	TRUE	FALSE	symptomatic
11311	ff49f6b7-f	2020-04-1	0.954	41.4	60.4	17	female	FALSE	TRUE	symptomatic
11312	ff5f97db-9	2020-04-1	0.7858	48.9	2.3	49	male	FALSE	TRUE	symptomatic
11313	ffbeb867-(	2020-04-2	0.9647			30	male	FALSE	FALSE	symptomatic
11314	ffe13fcf-c5	2020-04-1	0.9485	41.1	28.8	31	male	FALSE	FALSE	symptomatic
11315	fff3ff61-23	2020-06-2	0.5257	51.6	-0.2		female	FALSE	FALSE	symptomatic
11316	0012c608	2020-04-1	0.0482	-16.5	-71.5					
11317	001c85a8-	2020-04-1	0.0735	40.6	-3.6					
11318	00273cdf-	2020-05-1	0.0307	43.6	-6.9					
11319	002db0bd	2020-04-1	0.9536							
11320	0033d1d5	2020-07-2	0.9712							
11321	00343395	2020-04-2	0.824							
11322	00367af5-	2020-04-1	0.3009							

Figure 1: Excel sheet sorted from A-Z under status column showing empty status fields from row 11315 onwards

Because of this, almost half of our dataset is unusable given that it does not have a status.

2) Project Goal

For this project, our goal was to classify the COVID-19 status of an individual. Therefore, we were only interested in two statuses; whether the individual has COVID-19, or they are healthy. Because of this, we did not use the data which were labelled as symptomatic. As seen in Figure 2, row 9574 onwards contains data labelled as symptomatic.

1	uuid	datetime	cough_detected	latitude	longitude	age	gender	respiratory_condition	fever_muscle_pain	status
9568	ffdf80c2-b	2020-04-0	0.6991			26	male	FALSE	FALSE	healthy
9569	ffe5e2a4-e	2020-04-2	0.5591	13	77.6	26	male	FALSE	FALSE	healthy
9570	ffedc843-b	2020-06-0	0.9498	-34.5	-58.5	23	male	FALSE	FALSE	healthy
9571	ffeea120-9	2020-05-0	0.9784	14.3	121.1	22	female	FALSE	FALSE	healthy
9572	fff13fa2-a7	2020-04-1	0.7154	31.9	34.7	21	male	TRUE	FALSE	healthy
9573	fffaa9f8-4(	2020-04-1	0.0243	41	28.8	50	male	TRUE	TRUE	healthy
9574	0029d048-	2020-07-1	0.9456			35	male	TRUE	FALSE	symptomatic
9575	00432f00-	2020-04-3	0.6911	48.9	2.6	31	female	TRUE	FALSE	symptomatic
9576	006d8d1c-	2020-04-2	0.2677			28	male	TRUE	TRUE	symptomatic
9577	008c1c9e-	2020-04-1	0.9751	48.9	2.7	44	male	FALSE	FALSE	symptomatic
9578	00ce5b06-	2020-04-1	0.99	5.1	-73.6	41	female	TRUE	FALSE	symptomatic
9579	0160d81f-(	2020-04-1	0.9349	6.9	80	34	male	FALSE	TRUE	symptomatic
9580	01614a4a-	2020-04-2	0.938			23	male	FALSE	FALSE	symptomatic
9581	01ea827e-	2020-06-2	0.0163			30	male	FALSE	TRUE	symptomatic

Figure 2: Excel sheet sorted from A-Z under status column showing symptomatic status from row 9574 onwards

After truncating the data which are labelled as symptomatic, we are left with over 9,500 audio files to work with.

3) Imbalanced dataset

While analysing the remaining data, we realised that there was a significantly larger proportion of data labelled as healthy as compared to data labelled as COVID-19. This means that if we were to use the full dataset to train the model, it may develop a bias towards predicting audio samples as healthy. This would thus reduce the overall model performance in predicting if a given cough audio sample is indeed a COVID-19 class. Moreover, it would also skew the overall performance of the model as if the model is capable of predicting audio samples labelled as healthy, it would achieve a high accuracy regardless of how badly it performs in predicting audio samples labelled as COVID-19. As shown in Figure 3, cough audio samples labelled as COVID-19 barely exceed 1000 samples, meaning that there are over 8 times more healthy audio samples compared to COVID-19 audio samples.

1	uuid	datetime	cough_detected	latitude	longitude	age	gender	respiratory_condition	fever_muscle_pain	status
1006	feb52c0e-	2020-04-1	0.3081			51	male	FALSE	FALSE	COVID-19
1007	fee168b1-	2020-04-1	0.5634	41.9	60.2		female	FALSE	TRUE	COVID-19
1008	ff68cc55-6	2020-05-2	0.5984	-13	-38.5		female	FALSE	FALSE	COVID-19
1009	ff8363d2-(	2020-07-1	0.9933				female	FALSE	FALSE	COVID-19
1010	ffbca476-8	2020-04-1	0.7466	41	28.6		female	FALSE	FALSE	COVID-19
1011	ffe0658f-b	2020-04-1	0.9846	41.6	60.9	22	male	TRUE	TRUE	COVID-19
1012	00039425-	2020-04-1	0.9609	31.3	34.8	15	male	FALSE	FALSE	healthy
1013	0009eb28-	2020-04-1	0.9301	40	-75.1	34	male	TRUE	FALSE	healthy
1014	001328dc-	2020-04-1	0.9968			21	male	FALSE	FALSE	healthy
1015	0028b68c-	2020-05-2	0.8937			28	female	FALSE	FALSE	healthy
1016	00291cce-	2020-04-1	0.9883	39.4	67.2	15	male	FALSE	FALSE	healthy
1017	002d28bc-	2020-04-1	0.9959			46	female	FALSE	FALSE	healthy
1018	0037f67c-(	2020-04-1	0.8109	41.1	28.8	39	other	FALSE	FALSE	healthy
1019	003e1bf6-:	2020-04-1	0.9794			51	male	FALSE	FALSE	healthy
1020	0044cb7b-	2020-05-1	0.9947	41.6	-4.8	41	female	TRUE	FALSE	healthy

Figure 3: Excel sheet sorted from A-Z under status column showing only 1010 rows of COVID-19 data points

Because of this, we used only a subset of the audio samples classified as healthy to achieve a more balanced dataset. The list of audio data used and their respective labels can be found in Dataset.xlsx

4) Manual Quality Check

When listening to the remaining audio samples in the dataset to verify the integrity of the data, we realised that the dataset contained many samples which are not coughs at all. Given that the dataset is crowdsourced, a good proportion of the audio samples were contaminated with low-quality or irrelevant audio samples which did not represent coughs. For example, some of the audio samples contained only pure silence. Others simply contained sounds of people talking, singing, or making random noises. Because of this, we decided to manually sift through the data to locate samples which are coughs, keeping any samples which we felt were of good quality and sounded like a cough. We split the remaining 1,000 audio samples labelled as COVID-19 into three to be processed by each member. We also had each member in the team process an equivalent number of healthy audio samples. We manually marked in an excel sheet if an audio clip sounded like a cough after listening to it. Finally, we used a simple Python script to filter out only the data points which we labelled. Overall, we ended up with 1,645 audio samples for the final dataset after preprocessing.

After manually processing the audio files, we created a new excel sheet containing the metadata for the remaining audio files. This new metadata is stored in Dataset.xlsx.

4. System Architecture

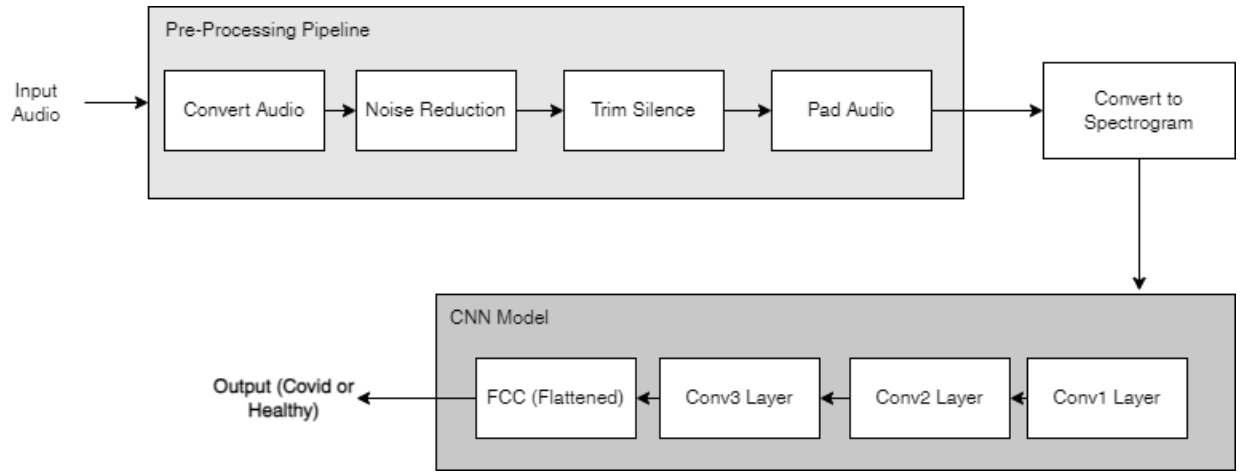

Figure 4: Overview of the model architecture

Our Covid Detection systems involve 2 components, a data pre-processing pipeline, and the CNN model itself. After the audio is processed, it is converted to a spectrogram, which is then sent to the CNN model for training/evaluation.

4.1 Data Processing Pipeline

4.1.1 Audio Conversion

The dataset consisted of audio clips recorded in .webm, .wav and .ogg formats. Moreover, some of the audio clips were recorded in stereo, while others were recorded in mono. This means that some of the audio files contain 2 audio channels instead of 1. As the neural network requires inputs of a consistent dimension, we had to standardise all the audio clips before converting them into a spectrogram and feeding the spectrogram into the model. We converted all the audio clips into .wav as it is a common audio format supported by most libraries, and we changed all the audio clips to a single channel as we realised most of the audio clips were recorded in mono while we were filtering the data. We also standardised the sampling rate for all the audio clips. Standardising the sampling rate would ensure that all audio samples fed into the network have the same time resolution, meaning that each unit of time in the audio data corresponds to the same duration across all samples. This is important as the time can also be seen as a feature, and inconsistencies in the time resolution might affect the model's ability to learn.

4.1.2 Noise Reduction

We found that a lot of audio data had varying amounts of background noise. While this would be good if there was sufficient data for training a more robust model, the dataset was too small. Thus, we decided to remove background noise from all our data.

4.1.3 Silence Removal

The audio data in the dataset contains numerous areas of silence before and after the cough. While manually filtering the dataset, we realised that the cough sounds did not necessarily start at the beginning of the audio clip. Some clips had cough sounds only near the end of the audio clip, and some clips had cough sounds in the middle of the audio clip. Because of this, we decided to remove silent regions from the audio clip after noise reduction.

This was done to ensure consistency in the positioning of cough sounds across all audio clips, facilitating a more accurate analysis and processing of the dataset. By standardising the start time of cough sounds to the beginning of each audio clip, it would simplify the training process for the models and potentially enhance the effectiveness of the model during classification or detection.

4.1.4 Audio Padding

Lastly, we also trimmed and padded all the audio clips to be exactly 10 seconds long. The reason we chose 10 seconds is because that is the maximum length of the audio clips in the dataset as described in the paper for the corpus. Moreover, standardising the length of the audio clip would be imperative in ensuring that our spectrogram images have a consistent dimension. After preprocessing the audio files, we can finally use them to train our model by converting them into spectrograms.

4.1.5 Converting Audio to Mel Spectrograms

As humans perceive sound on a logarithmic scale instead of a linear one, the model may train better on Mel spectrogram data, as neural networks resemble the human brain (Zhang et al., 2019). Therefore, the inputs are converted to a Mel spectrogram using the built-in torch functions and the amplitude is then converted to decibels.

To plot the original waveform, some data manipulation is performed to turn the waveform into a numpy, and the time window is calculated by dividing the number of frames with the sample rate.
Librosa is used to visualise the Mel spectrogram.

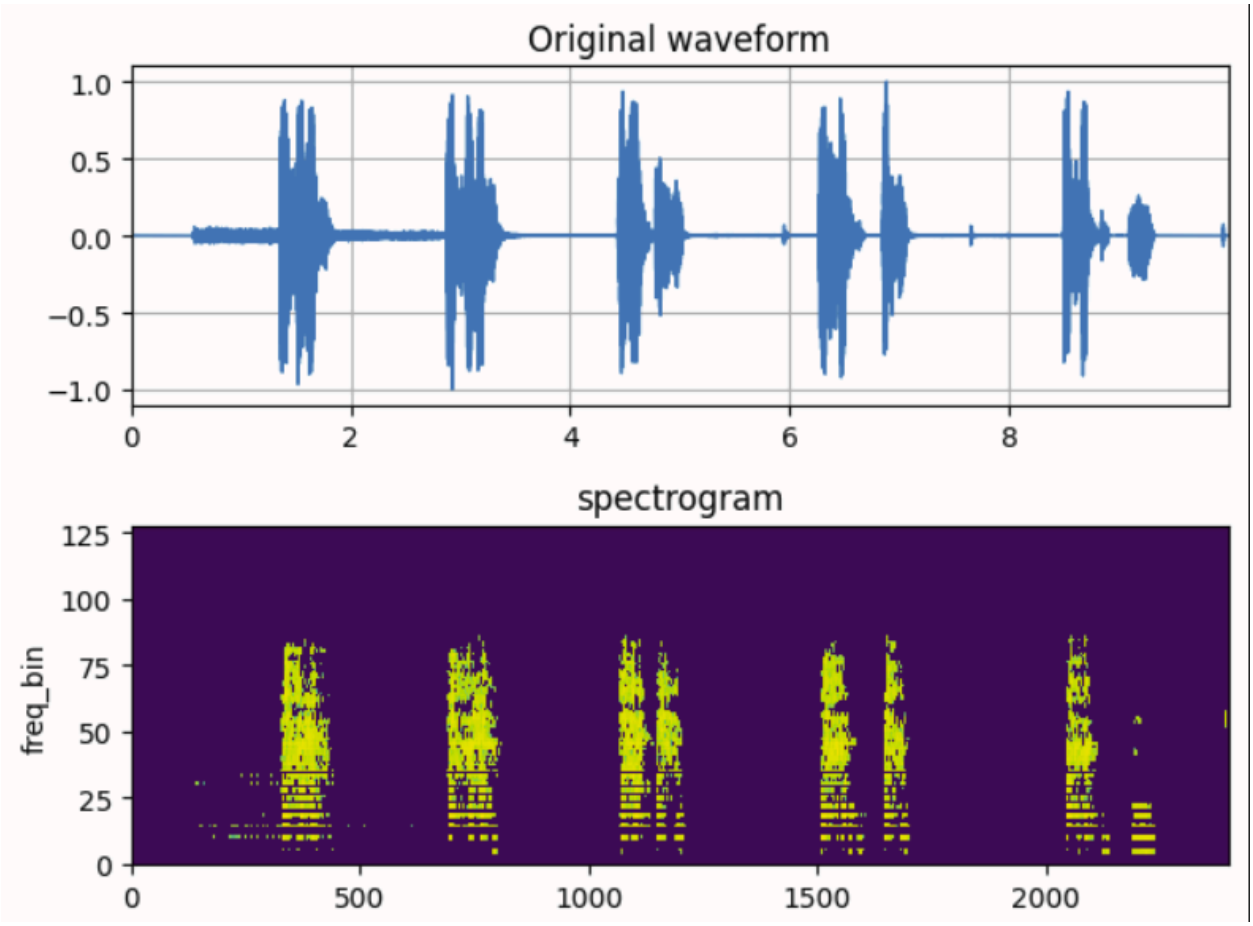

Figure 5: An example of a visualised datapoint

4.2 Model Architecture

Our model consists of 3 convolution blocks and a fully connected linear layer. Each convolution block contains the following:

- A 2D convolution layer
- A ReLU function
- Batch Normalisation
- Max Pooling

The linear layer takes in the spectrogram that is produced by the convolution layers and outputs the probability of whether the cough comes from a COVID or healthy individual.

4.3 Saving and loading models

The model saves the best latest model with the highest validation accuracy with the name "model_iter{epoch}", where the epoch is the training iteration with the best validation accuracy.

5. Results

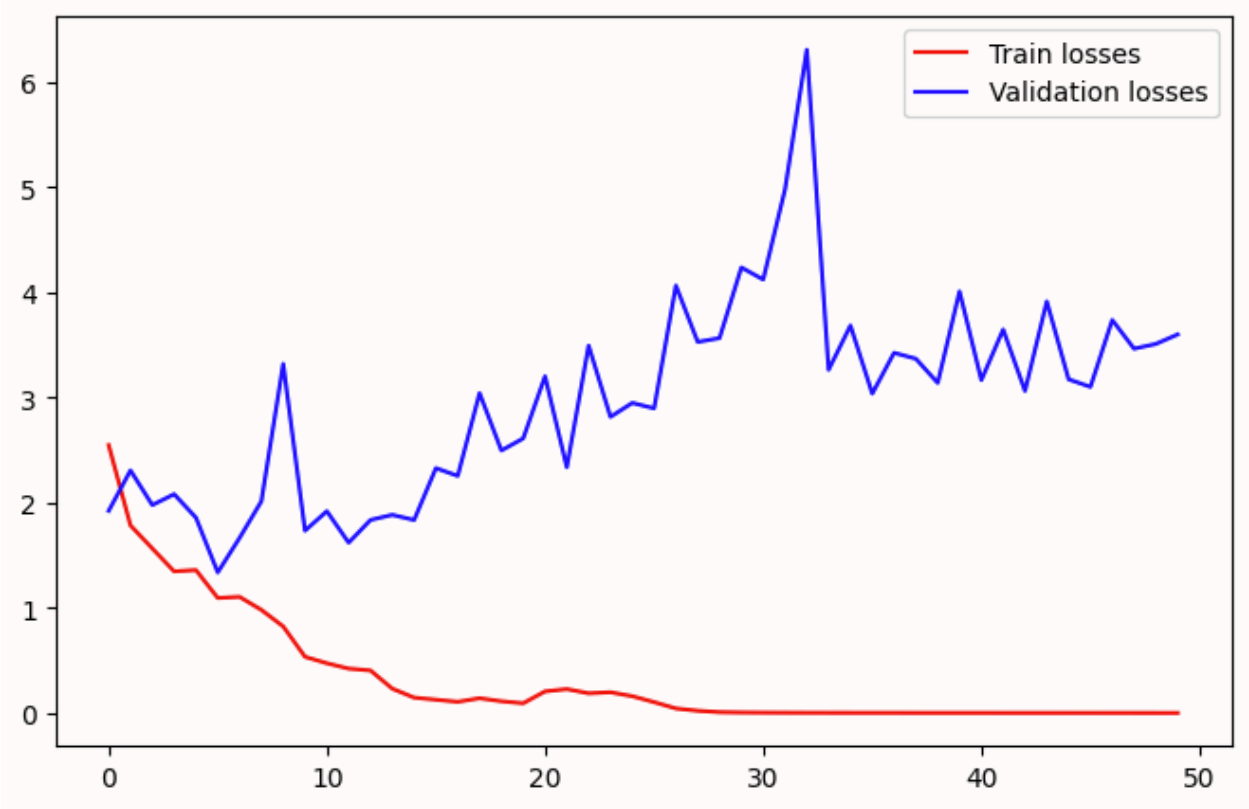

Figure 6: Graph of Training and Validation Loss

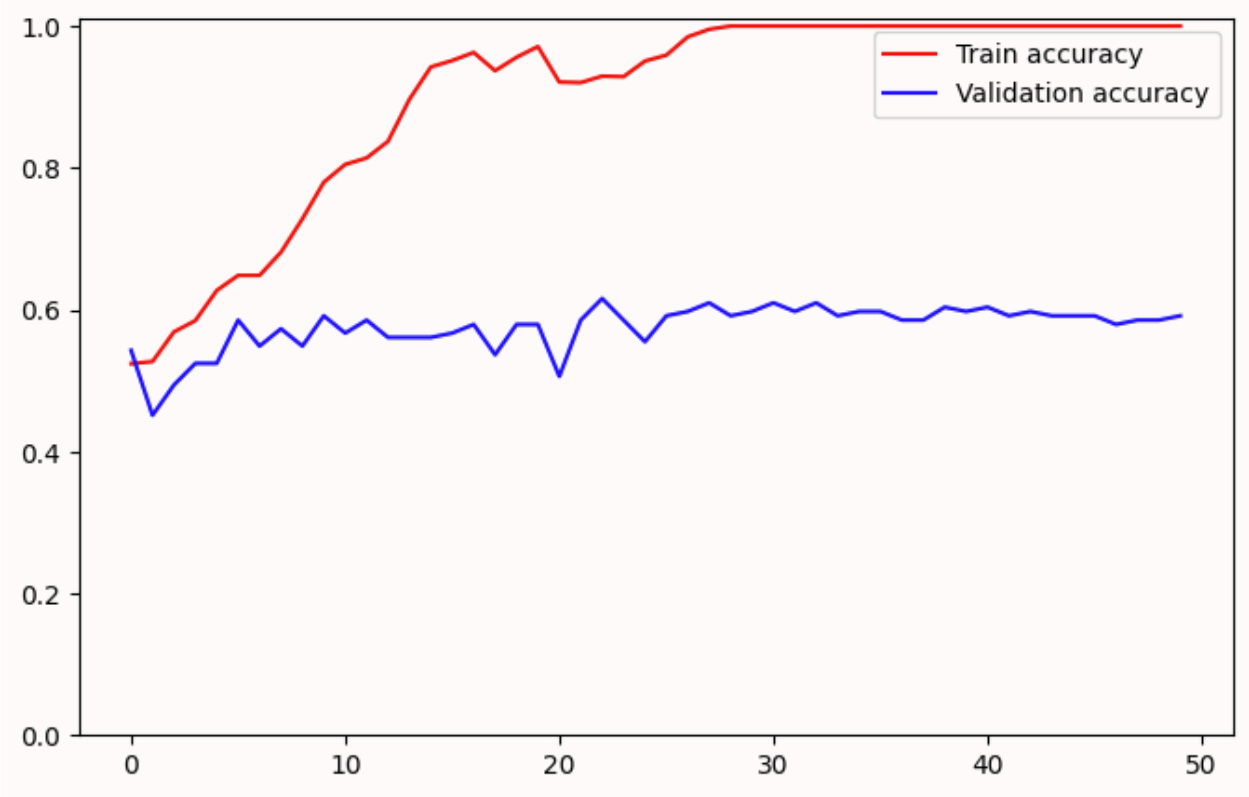

Figure 7: Graph of Training and Validation Accuracy

6. Challenges

As the model was performing poorly, it seems to be overfitting. We tried numerous methods to improve the model and rectify the overfitting issue, however we could not get the model to perform much better than a coin flip. An example of another model that determines whether a patient has COVID using cough audio and other features as input produces a 75.54% accuracy (Pentakota, 2019), which is significantly better than our model. A very likely reason for this is the much smaller dataset which we were working with. Our model could be improved by introducing more features for the model to learn from, and sourcing from much larger datasets.

7. Conclusion

Training a model that uses audio to produce highly accurate predictions of symptomatic patients is a difficult and time consuming task, which takes many layers and trial and error to increase the accuracy. While through our efforts, the results of our model was disappointing, we believe that if we had a better

dataset and more time to experiment, it would have been possible to improve our model further and obtain better results.

8. Project Contributions

Name	Project Contributions
Lee Chang Zheng	Creating the main code for the model, dataset and dataloader
Lee Cheng Xin	Split Data and created Data Processing Pipeline
Jason Peng Jing Ming	Created visualisations and saver loader code. Main runner of the model
Everyone	Manually processed over 600 audio samples, half of which are audio samples labelled as COVID-19 while the other are samples labelled as healthy.

7. References

https://pytorch.org/audio/main/tutorials/audio_feature_extractions_tutorial.html#sphx-glr-tutorials-audio-feature-extractions-tutorial-py

Zhang, B., Leitner, J., & Thornton, S. (2019). Audio Recognition using Mel Spectrograms and Convolution Neural Networks. https://www.semanticscholar.org/paper/Audio-Recognition-using-Mel-Spectrograms-and-Neural-Zhang-Leitner/6323d5a0748e2bc6e85e56b521450c4e03fa9082

Pentakota, P., Rudraraju, G., Sripada, N. R., Mamidgi, B., Gottipulla, C., Jalukuru, C., Palreddy, S. D., Bhoge, N. K., Firmal, P., Yechuri, V., Jain, M., Peddireddi, V. S., Bhimarasetty, D. M., Sreenivas, S., Prasad K, K. L., Joshi, N., Vijayan, S., Turaga, S., & Avasarala, V. (2023). Screening covid-19 by SWAASA AI platform using Cough sounds: A cross-sectional study. *Scientific Reports*, *13*(1). https://doi.org/10.1038/s41598-023-45104-4