COVBOT: INTELLIGENT AI CHATBOT FOR COVID-19

A PROJECT REPORT BY

APAR GARG (A0231539E)
LIN QIANQIAN (A0231438J)
LIU JING (A0231403Y)
YANG MINGRUN (A0231344R)
ZHU JUNKUN (A0231471N)

SUBMITTED TO

INSTITUTE OF SYSTEMS SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE
21 LOWER KENT RIDGE RD, SINGAPORE 119077

in

Partial fulfillment of the requirements for the degree of

MASTER OF TECHNOLOGY

APRIL 2022

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Background

Coronavirus 2019 (COVID-19), an infectious disease caused by severe acute respiratory syndrome coronavirus type 2 (SARS-CoV-2), has caused an ongoing outbreak that has become one of the deadliest pandemics in human history. The first known case of the disease was confirmed in Wuhan, Hubei Province of the People's Republic of China, in late 2019, and has since been detected worldwide [1]. The World Health Organization (WHO) declared the COVID-19 outbreak a Public Health Emergency of International Concern (PHE IC) on 30 January 2020 and assessed IT to have pandemic characteristics on 11 March 2020. UN Secretary-General António Guterres has described the COVID-19 pandemic as the most serious crisis humanity has faced since World War II [2].

1.2. Problem

The ongoing pandemic is having a huge impact on all aspects of people's lives around the world. One of the main problems is that the worldwide reaction to COVID-19 has yielded a fast development of articles and scientific publications. This information can help people to understand the virus better. In such difficult times, finding answers on search engines like Google, despite them being the de-facto places to get answers to all queries, can be very frustrating and overwhelming since the information is spread across different web pages. No single web source can provide the answers to all queries. A solution is needed to help people get the answers to all kinds of COVID-19 related queries. Similarly, the outbreak of the COVID-19 has been accompanied by a large amount of misleading and false information about the virus, especially on social media. On February 15th, 2020, the World Health Organization (WHO) Director-General Tedros Adhanom Ghebreyesus said that “We’re not just fighting an epidemic, we’re fighting an endemic” [3]. In such a situation, a solution is needed to segregate fake news from authentic news. Another thing worth noting is that the gold standard for coronavirus testing is RT-PCR, which always takes a long time to produce results and costs a lot [4]. A cheaper and faster diagnosis solution is the need of the hour.

1.3. Solution

In this project, we have introduced a medical domain-specific chatbot system using signal processing, machine learning, deep learning, and natural language processing techniques. The system can answer frequently asked queries related to COVID-19 by gathering and summarizing information from various authentic web sources. Moreover, it provides other useful features including the ‘Fake tweet detection’ and ‘COVID-19 screening using cough audio recordings’.

2. FAQ ANSWERING

2.1. Introduction

People may have a lot of questions in their minds regarding symptoms, screening, treatment options, etc. Thousands of articles, blogs, and scientific publications are published every week about COVID-19, SARS-CoV-2, and related topics on the internet [5]. In such difficult times, finding answers on the internet to all queries can be very frustrating since the information is spread across different web pages. No single web source can provide the answers to all queries. In addition, with misinformation being rampant these days, checking the authenticity of information on web pages is next to impossible. So, to solve this problem, we have designed a COVID-19 question answering functionality, which helps to answer questions about COVID-19. The FAQ Answering functionality is a retriever-reader dual algorithmic approach that takes questions from the users as input and generates the most accurate responses consisting of a single line answer to the query, the title of the literature, and a whole paragraph from the scientific literature [6].

2.2. Dataset Description

In response to the COVID-19 pandemic, the White House and a coalition of leading research groups have prepared the COVID-19 Open Research Dataset (CORD-19). CORD-19 is a resource of over 59,000 scholarly articles, including over 47,000 with full text, about COVID-19, SARS-CoV-2, and related topics. This freely available dataset is provided to the global research community to apply recent advances in natural language processing and other AI techniques to generate new insights in support of the ongoing fight against this infectious disease [7].

2.3. Methodology

2.3.1. Dataset Preparation

After downloading the dataset, we examined the contents of the dataset. Since it was stored in JSON format, the structure was too complex to directly perform analysis.

Referring to the others' EDA work [8], we went through all the JSON files and did the data cleaning. Because of the follow-up QA, we only focused on the text message. So, we walked through the JSON file of the entire dataset and only extracted the title summary and text information for each article. Then we used this information to generate a CSV file for further work.

2.3.2. Related Articles Search

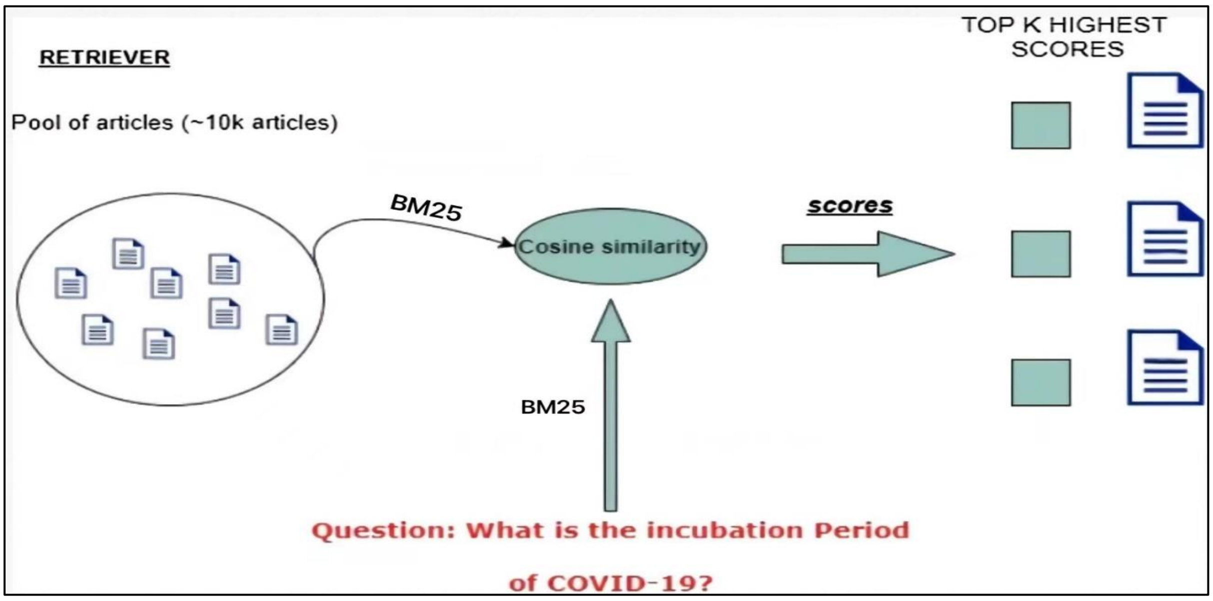

RETRIEVER
Pool of articles (~10k articles)
BM25
Cosine similarity
scores
TOP K HIGHEST SCORES
BM25
Question: What is the incubation Period of COVID-19?

Figure 1: Related articles search method

We needed to find articles from the article library most relevant to the user question. We could have used TF-IDF (term frequency-inverse document frequency) with cosine similarity to calculate the similarity degree between the user question and the articles. However, here we used one of the simplest and most classical algorithms—Okapi BM25 (BM is an abbreviation of best matching) [9] which is a ranking function used by search engines to estimate the relevance of documents to a given search query. M25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document. TF-IDF rewards term frequency and penalizes document frequency. BM25 goes beyond this to account for document length and term frequency saturation. Though this algorithm is very old, it still can perform a good job to give us the most relevant articles. Figure 1 shows the structure of the related articles search method.

2.3.3. Answer Retrieval

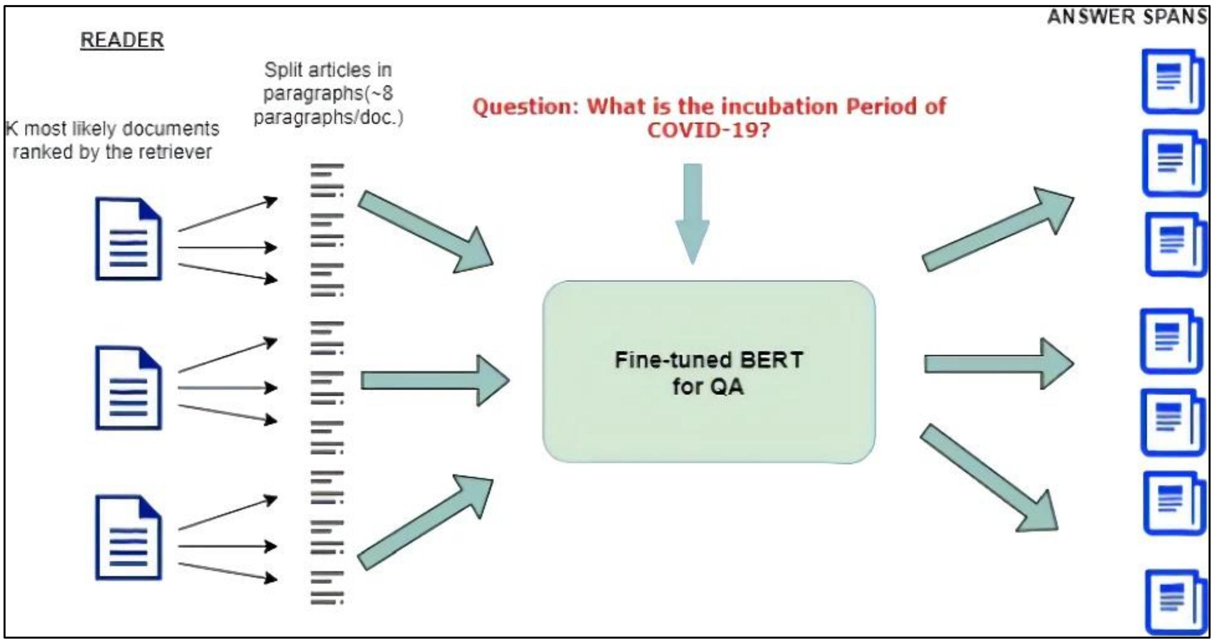

READER
Split articles in paragraphs(~8 paragraphs/doc.)
K most likely documents ranked by the retriever
Question: What is the incubation Period of COVID-19?
Fine-tuned BERT for QA
ANSWER SPANS

Figure 2: Answer retrieval method using BERT

The previous section gave us a list of articles. Now, for each article (context) and the same user query pair, we asked the pre-trained and pretty-powerful transformer model what is the part of the context that better represents the query. HuggingFace [10] is a library that permits us to work with ease with such complex and big neural networks. It provides a large number of pre-trained BERT (Bidirectional Encoder Representations from Transformers) models that can be fine-tuned and used for various tasks.

We chose the bert-large-uncased-whole-word-masking-finetuned-squad [11] model in this part. It is a pre-trained model of the English language using a masked language modeling (MLM) objective. This model is uncased: it does not make a difference between english and English. This model is using both “book corpus” and “Wikipedia” datasets to do training. Then it was fine-tuned on the SQuAD dataset. The model performed well when migrating to novel coronavirus-related QA tasks. So, we decided not to fine-tune it further, but to go straight with the pre-trained model. With the help of this model, we easily obtained the answer to the question of where the tokens start and end.

In order to provide quick answers to the user, our FAQ Answering functionality should extract answers from documents in real-time. So, we decided to store the (question, extracted answer) pairs in a CSV file and use it during inference in place of real-time answer extraction. Figure 2 shows the answer retrieval method using BERT.

2.3.4. Question Similarity Comparison

We prepared 38 relevant questions in advance and successfully generated a question-answer list by using the above two-part model.

Now, for a user's input query, we needed to find the most similar question existing in our question list to give the answers [12]. We used another pre-trained sentence transformer model (bert-base-nli-mean-tokens) here to generate the sentence vectors that can be used to compare similarity. Then we used the cos-similarity to find the most similar question and return its corresponding answer.

However, due to our limited number of questions, we could not necessarily cover all possible user questions. Therefore, we determined the boundary value for the system. When the similarity of the most similar question is found to be higher than 0.75, we assume it to be consistent with the question asked by the user. When the value is between 0.75 and 0.6, we will show the question to the user and ask if it is the same question. Otherwise, we simply tell the user that the question he is asking may not exist in our database.

2.4. Results

Table 1: Performance of FAQ Answering functionality

	It's the exact answer	**Relevant but inaccurate**	**Irrelevant to the question**	**Proportion of relevant answers**
Key answer from the matched paragraph	12	9	9	0.7
Matched paragraph	20	4	6	0.8

To evaluate the performance of our FAQ Answering functionality, we collected 30 questions about COVID-19 from the internet, including how it is transmitted, how it can cause infection, and how to protect against it. The model provides both the relevant matched paragraph and the key answers in the paragraph. We put these two sections on a manual scale and divided them into three criteria: accurate, relevant but not accurate, and completely irrelevant. Table 1 shows the performance of the proposed FAQ Answering functionality based on the criteria mentioned above.

As can be seen from the table, the accuracy of the matched paragraph and the key answer is 0.8 and 0.7 respectively (because the complete answer paragraph contains the key answer, but the key answer may not be accurately screened). Considering that the dataset consisted mostly of papers from medical institutions, rather than articles popularizing COVID-19 knowledge, we believe the accuracy of our FAQ Answering functionality is acceptable.

3. FAKE TWEET DETECTION

3.1. Introduction

The fake news about COVID-19 is rampant on social media and various other media and is a matter of serious concern due to its ability to cause a lot of social and national damage [13]. However, the scope for detection is so limited because they depend on manual human detection. In a world with millions of tweets either removed or being published every minute, it is not feasible to annotate and segregate manually. So, to solve this problem, we have designed a fake tweet detection functionality. The functionality detects if a tweet related to COVID-19 is authentic or fake, by applying several machine learning algorithms and deep learning algorithms. Besides determining whether the given tweet is fake or not, our system can show which part of the tweet contributes most to the negative result by exploring the visualization of attention.

3.2. Dataset Description

Table 2: Number of tweets per class

Class	**Training dataset**	**Testing dataset**	**Total**
Real	3360	1120	4480
Fake	3060	1020	4080
Total	6120	2140	8240

The dataset [14] is an open-source labeled Kaggle dataset, which consists of a training dataset and a test dataset. Each CSV file includes three columns: id, tweet, and label. Some of the tweets provide real/authentic information while some do not. All the tweets in the dataset have been manually annotated with real or fake. The number of tweets per class in training and testing datasets is shown in Table 2.

3.3. Methodology

3.3.1. Dataset Preparation

To prepare the dataset, we applied a number of pre-processing techniques including encoding the categorical variables, cleaning the text (removing URLs, symbols, emojis, HTML tags, unnecessary blank text between words, duplicate data, etc.), lemmatization, and stop words removal.

3.3.2. Feature Extraction & Modeling

3.3.2.1. TF-IDF + ML Models

TF-IDF method was used to get word embeddings from tweets. TF-IDF is a common statistical measure to evaluate how relevant a word is to a document. We trained the Naive Bayes ML model on these TF-IDF embeddings for fake tweet detection. We also trained 2 DL models namely LSTM (Long short-term memory), and CNN (Convolutional Neural Network) [15].

3.3.2.2. Word2Vec + LSTM

Word2Vec method was used to get word embeddings from tweets. Word2Vec uses a neural network model to learn word associations from a large corpus of text. A vector is assigned to each unique word in that corpus. Words that share common contexts are positioned in proximity to one another in the vector space generated. We trained an LSTM model on these Word2Vec embeddings for fake tweet detection.

3.3.2.3. GloVe + LSTM

GloVe (Global Vectors) method was used to get word embeddings from tweets. GloVe uses a neural network model to learn word associations from a large corpus of text. A vector is assigned to each unique word in that corpus. Words that share common contexts are positioned in proximity to one another in the vector space generated. We trained an LSTM model on these GloVe embeddings for fake tweet detection.

3.3.2.4. Fine-tuning BERT

BERT was fine-tuned to get word embeddings from tweets and classify the tweets into real or fake. BERT is a transformer-based pre-trained model released by google. The advantage is that the BERT model has very stable and good performance for solving most NLP problems through numerous corpus training. The features extracted by BERT are more stable and reliable. But for domain-specific tasks, BERT may not be customized enough, so we use our dataset to fine-tune the BERT classifier.

3.3.2.5. Early Fusion

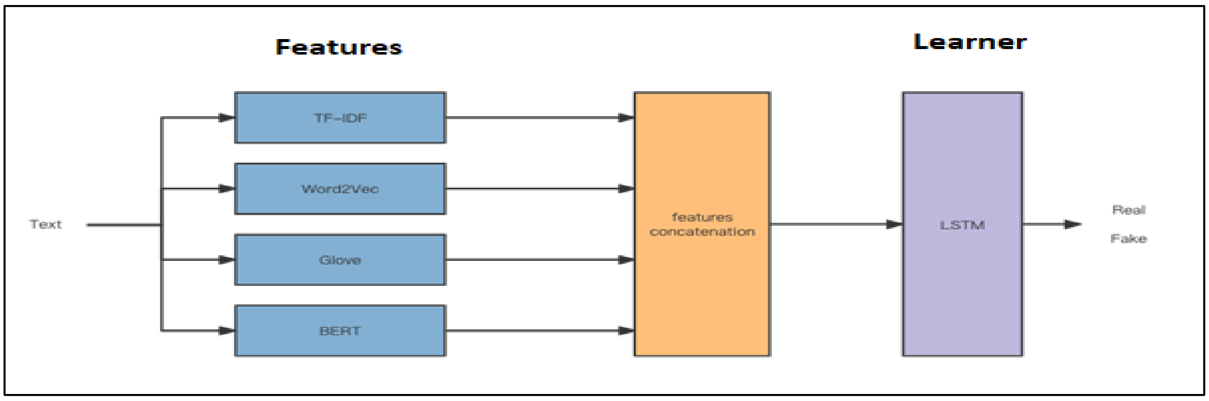


Figure 3: Proposed Early Fusion Model Architecture for Fake Tweet Detection

Word embeddings from all methods (TF-IDF, Word2Vec, Glove, and BERT) were concatenated to get new word embeddings. We trained an LSTM model on these concatenated embeddings for fake tweet detection. The proposed model architecture for early fusion is shown in Figure 3.

3.3.2.6. Late Fusion

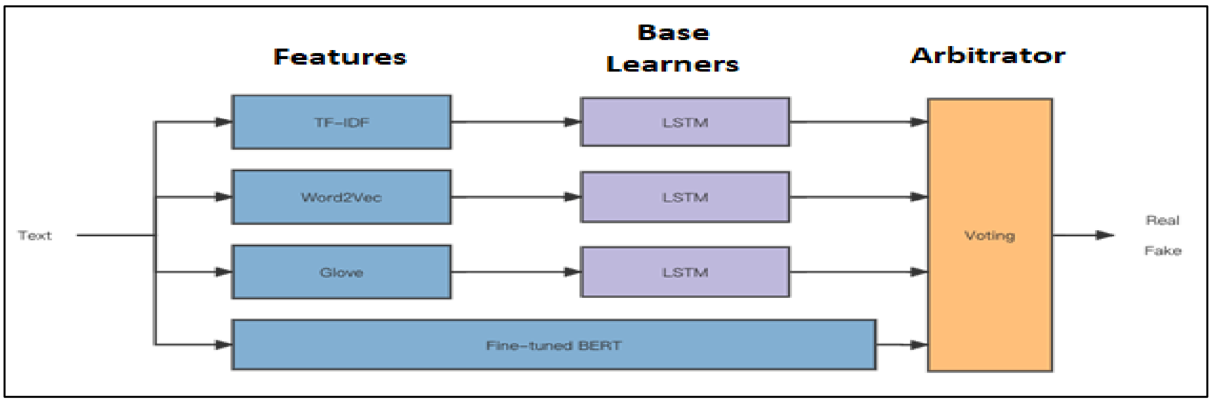


Figure 4: Proposed Late Fusion Model Architecture for Fake Tweet Detection

4 models were chosen for the ensemble. All models (base learners) output real or fake. The system takes in these predictions from the base learners and outputs the class which has the highest number of votes with a weighted scheme. This method is known as Weighted Hard Voting. The proposed model architecture for late fusion is shown in Figure 4.

3.3.3. Fake Information Extraction

We designed a method to highlight parts/words of the fake tweets that hold the wrong/fake facts. Our visualization method is based on the fine-tuned BERT model. We design the visualization method by analyzing the documentation of BertForSequenceClassification [16] and referring to the implementation of BertViz [17]. By parsing the attention information, we had twelve layers of attention data. Each layer had a shape of (128, 128), representing the attention between each token. For each layer, we first traversed each Token A, found the Token B with the highest attention, and then counted the Token B. It was believed that the higher the number of appearances of Token B, the higher the attention of more tokens to it, and the higher importance it contributes to the final result.

3.4. Results

Table 3: Overall performance of Fake Tweet Detection on test data

Model	**Training Accuracy**	**Testing Accuracy**	**Testing Precision**	**Testing Recall**	**Testing F1 score**
TF-IDF+ Naive Bayes	0.91	0.91	0.92	0.93	0.91
TF-IDF+ CNN	0.87	0.88	0.87	0.84	0.87
TF-IDF+ LSTM	0.87	0.87	0.88	0.86	0.87
Word2vec+ LSTM	0.84	0.83	0.84	0.83	0.85
GloVe+ LSTM	0.87	0.84	0.86	0.85	0.85
Fine-tuned BERT	0.92	0.95	0.95	0.93	0.94
Early fusion	0.76	0.75	0.76	0.76	0.75
Late fusion	0.89	0.89	0.89	0.88	0.87

Table 4: Performance of Fake Information Extraction functionality

Dataset Type	**Completely correct**	**Correct with missing or over extraction**	**Completely incorrect**
Test dataset	6	10	4

The results of all our fake tweet detection experiments are listed in Table 3.

For fake information extraction, we manually annotated fake parts/words in 20 fake tweets in the test dataset. To evaluate the effectiveness of our proposed method, we classify the annotations into three levels: completely correct, correct with missing or over extraction, completely incorrect, and perform. The results are shown in Table 4.

4. COVID-19 SCREENING USING COUGH

4.1. Introduction

COVID-19 has been found to cause serious sickness and mortality in 2% to 3% of those infected over the world, particularly among the elderly and those with prior medical conditions or weakened immune systems [18]. Early diagnosis and treatment give patients a significantly higher chance of survival. In addition, infected people are contagious for 10 days on average, and they can spread the virus even if they don't show any symptoms. Early diagnosis lowers the rate of transmission, preventing or slowing the spread of an epidemic. [19]

To date, the gold standard for coronavirus testing is reverse transcription-polymerase chain reaction (RT-PCR) [20]. However, the RT-PCR test is quite unpleasant, needs person-to-person contact to administer, and takes a long time to produce results. Moreover, this test is not yet accessible to the people living in remote areas, where medical facilities are scarce [21]. Even though governments throughout the world have launched a free massive testing campaign to stop the virus from spreading, it is costing them billions of dollars each day at an average charge of $23 per test [22]. Hence, testing that is easily accessible, quick, and affordable is crucial to curb the spreading of the virus.

Dry cough is one of the more prevalent symptoms of respiratory tract infections, occurring in 68 % to 83 % of individuals who come in for a medical check [23]. In this project, we have built an automated diagnosis system based on machine learning, deep learning, and signal processing techniques to predict the presence of COVID-19 using the subject's cough audio recording. First, our system extracts various acoustic features from the subject's cough audio sample using various signal processing techniques. Then, various machine learning and deep learning algorithms use these acoustic features to classify the sample as COVID-19 or Non-COVID-19.

4.2. Dataset Description

Table 5: Number of audio samples per class in the Virufy dataset

Class	Number of samples
COVID-19	48
Non-COVID-19	73
Total	121

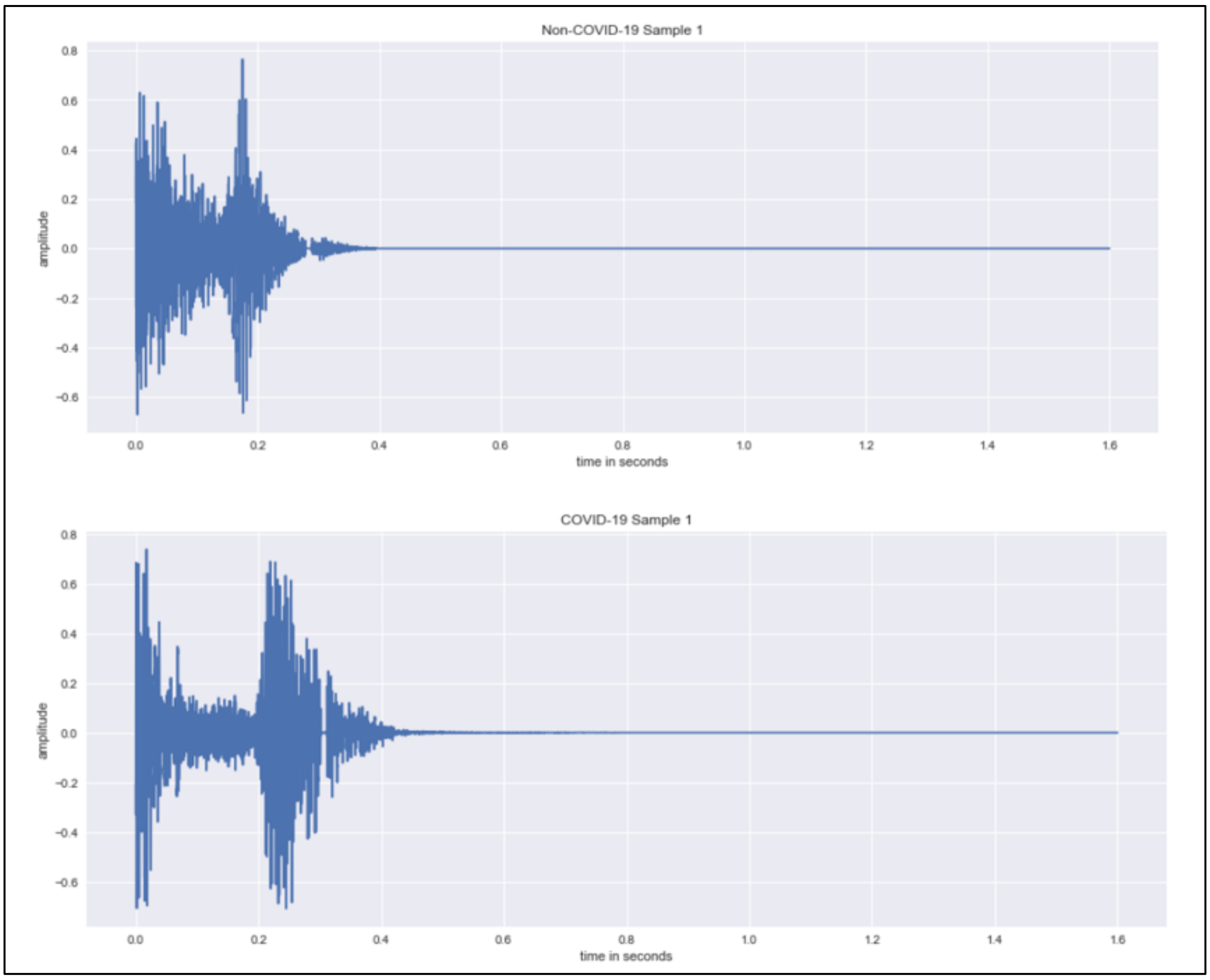


Figure 5: Comparison of the cough sounds for a Non-COVID-19 subject and a COVID-19 patient in the Virufy database

Virufy COVID-19 Open Cough Dataset [24] is an open-source dataset. It consists of 121 segmented cough samples from 16 patients. The audio samples have been donated by patients from multiple countries. Table 5 shows the number of segmented audio samples per class in the dataset. As evident from Figure 5, both the intermediate and voiced phases are longer for a COVID-19 subject than for a Non-COVID-19 subject. Moreover, the signal amplitude during the voiced phase is higher for a COVID-19 subject than for a Non-COVID-19 subject. The differences mentioned above indicate that the cough sound can be used as a valuable tool to discriminate the COVID-19 subject from a Non-COVID-19 subject.

4.3. Methodology

4.3.1. Data Preparation

Table 6: Number of audio samples generated per class

Class	Number of samples
COVID-19	652
Non-COVID-19	842
Total	1494

A number of techniques were used to pre-process and augment the audio samples. Each segment has a sampling frequency of 48000 Hz. First, we resampled the audio file to 16000 Hz. Then, silence removal was performed by extracting the signal envelope and thresholding it. Later, we segmented and augmented the resultant audio files. For this, we chose a random audio sample from the dataset according to their probability distribution (based on the duration of audio files) and then cropped a random duration of 1/10 sec from the chosen audio sample. We did this 1494 times (2 * audio samples of duration 1/10 secs taken from all audio files). Table 6 shows the number of audio samples generated per class after data preparation.

4.3.2. Feature Extraction & Modeling

4.3.2.1. MFCC + LSTM

From every 1/10 sec audio segment, we extracted 13 MFCCs (2D array). Then we trained an LSTM model on these features. 80% of data was used for training and the rest 20% was used for testing/validation.

4.3.2.2. MFCC + 2DCNN

From every 1/10 sec audio segment, we extracted 13 MFCCs (2D array). Then we trained a 2DCNN model on these features. 80% of data was used for training and the rest 20% was used for testing/validation.

4.3.2.3. Mean, Std of Acoustic Features + ML Models

Table 7: List of extracted acoustic features for modeling

Category	**Feature (Mean, std of …)**
Time domain	zcr (zero crossing rate), energy, energy entropy, delta zcr, delta energy, delta energy entropy
Frequency/Time-frequency domain	spectral centroid, spectral spread, spectral entropy, spectral flux, spectral rolloff, mfcc 1, mfcc 2, mfcc 3, mfcc 4, mfcc 5, mfcc 6, mfcc 7, mfcc 8, mfcc 9, mfcc 10, mfcc 11, mfcc 12, mfcc 13, chroma 1, chroma 2, chroma 3, chroma 4, chroma 5, chroma 6, chroma 7, chroma 8, chroma 9, chroma 10, chroma 11, chroma 12, chroma std, delta spectral centroid, delta spectral spread, delta spectral entropy, delta spectral flux, delta spectral rolloff, delta mfcc 1, delta mfcc 2, delta mfcc 3, delta mfcc 4, delta mfcc 5, delta mfcc 6, delta mfcc 7, delta mfcc 8, delta mfcc 9, delta mfcc 10, delta mfcc 11, delta mfcc 12, delta mfcc 13, delta chroma 1, delta chroma 2, delta chroma 3, delta chroma 4, delta chroma 5, delta chroma 6, delta chroma 7, delta chroma 8, delta chroma 9, delta chroma 10, delta chroma 11, delta chroma 12, delta chroma std

Table 8: Feature Selection techniques applied for dimensionality reduction

Category	**Method**
Filter	Pearson's Correlation, Variance Threshold
Wrapper	Recursive Feature Elimination (RFE)

```
mfcc_1_mean,mfcc_2_mean,mfcc_3_mean,mfcc_4_mean,mfcc_5_mean,mfcc_
6_mean,mfcc_7_mean,mfcc_8_mean,mfcc_9_mean,mfcc_10_mean,mfcc_11
_mean,mfcc_12_mean,mfcc_13_mean,delta mfcc_1_mean,delta mfcc_2
_mean,energy_entropy_std,spectral_entropy_std,mfcc_1_std,mfcc_2
_std,delta spectral entropy std
```

Figure 6: Top 20 features selected using feature selection techniques

From every 1/10 sec audio segment, we extracted 68 mixed domain (time domain and frequency/time-frequency domain) audio features. The list of extracted features is provided in Table 7. 80% of the data was used for training and the rest 20% was used for testing. We used feature selection to shortlist the top 20 useful features. The list of feature selection techniques used is provided in Table 8 and the list of 20 selected features is shown in Figure 6. Then we trained various ML models on these features. ML models namely: Naive Bayes, Random Forest, Gradient Boosting, Logistic Regression, KNN, and SVM.

4.4. Results

4.4.1. MFCC + LSTM

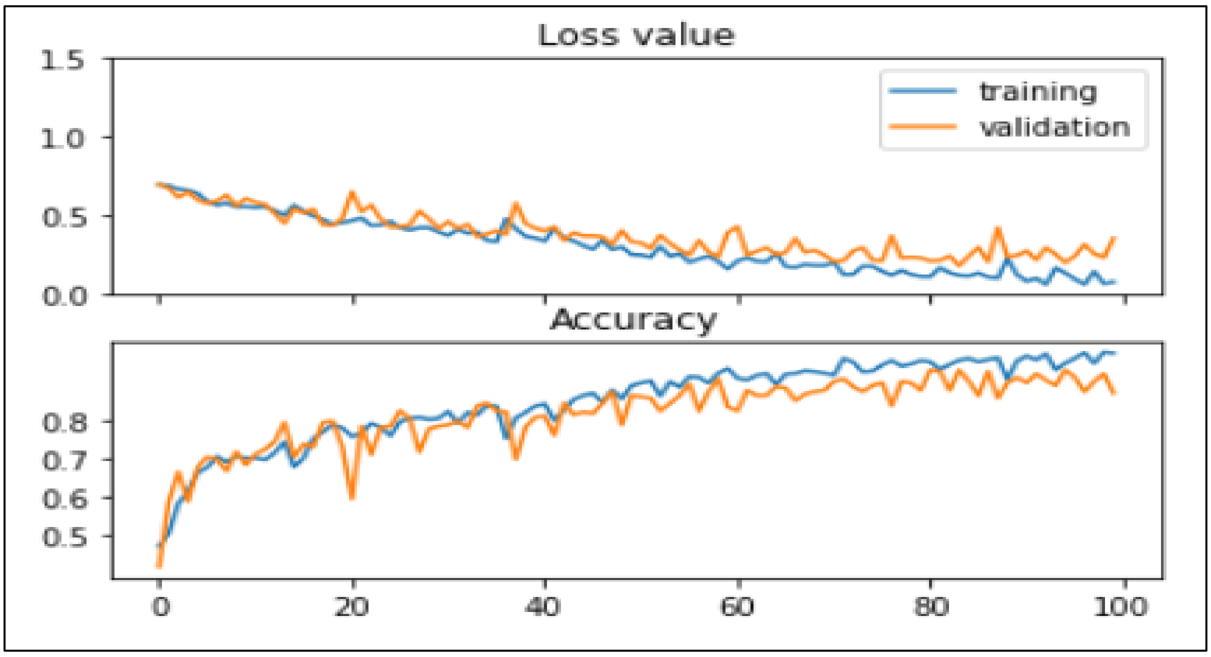


Figure 7: Loss and accuracy curves for LSTM trained on MFCC features

LSTM model trained on MFCC features achieved an accuracy of approximately 0.97 and 0.92 on training and validation datasets respectively. Figure 7 shows the loss and accuracy curves for the model. We also tested the model on 4 additional cough audio files (two files for COVID-19 positive and two files for COVID-19 negative) taken from Coughvid [26] dataset, and the model classified all 4 cases correctly.

4.4.2. MFCC + 2DCNN

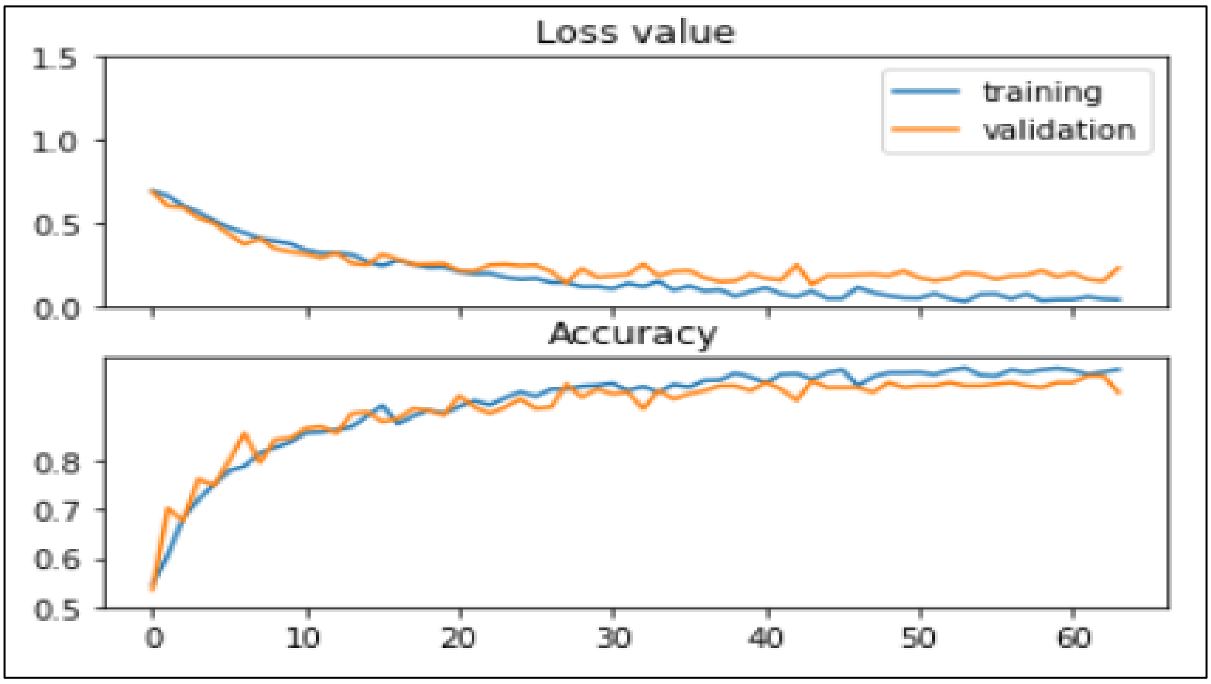


Figure 8: Loss and accuracy curves for 2DCNN trained on MFCC features

2DCNN model trained on MFCC features achieved an accuracy of approximately 0.98 and 0.94 on training and validation datasets respectively. Figure 8 shows the loss and accuracy curves for the model. We also tested the model on 4 additional cough audio files (two files for COVID-19 positive and two files for COVID-19 negative) taken from Coughvid [26] dataset, and the model classified all 4 cases correctly.

4.4.3. Mean, Std of Acoustic Features + ML Models

Table 9: Overall performance of COVID-19 Screening Using Cough on test data

Model	Training Accuracy	Testing Accuracy	Testing Precision	Testing Recall	Testing F1 score
Naive Bayes	0.65	0.64	0.69	0.60	0.57
Random Forest	1.00	0.90	0.90	0.90	0.90
Gradient Boosting	1.00	0.89	0.89	0.89	0.89
Logistic Regression	0.68	0.68	0.71	0.65	0.64
KNN	0.96	0.92	0.92	0.91	0.92
SVM	0.57	0.57	0.29	0.50	0.37

The results of all our ML models trained on the mean and std of various acoustic features are listed in Table 9.

5. SYSTEM ARCHITECTURE AND DEPLOYMENT

5.1. UI Description

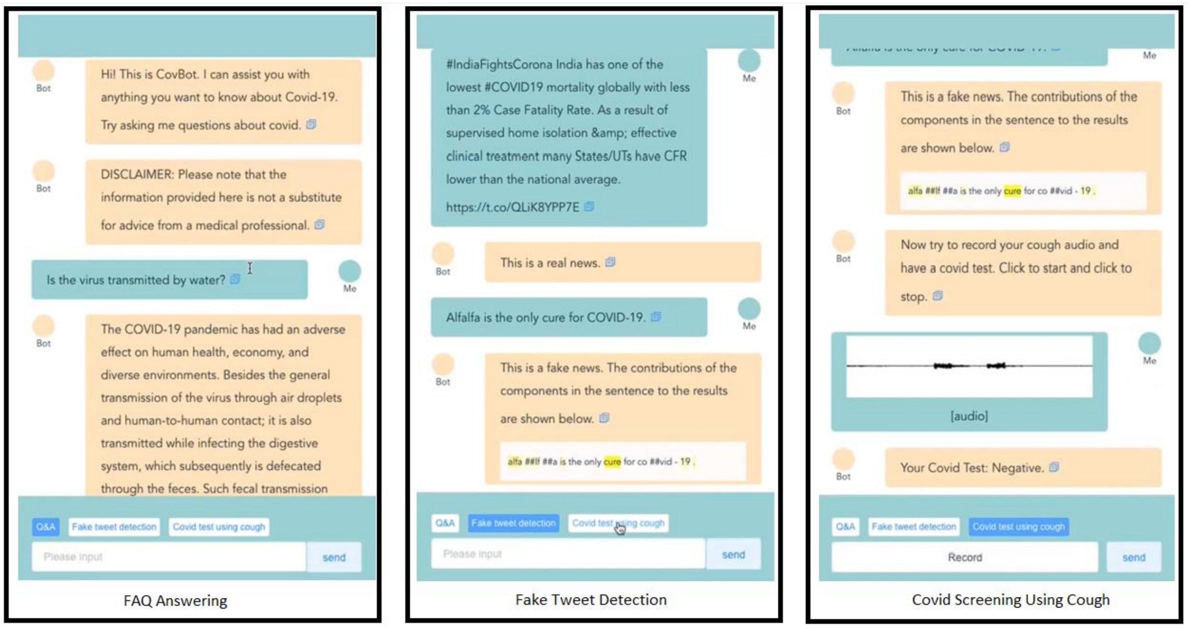
After the user enters the system, the chatbot will first have a welcome and disclaimer. The user can click the options button above the input box to switch between three different functions, the default is FAQ Answering. For all text information, users can click the icon at the end of the text to translate it into Chinese.

For FAQ Answering, users can enter any question about COVID-19 they want to ask, and the chatbot will analyze and search the library to see if there is a known answer. If there is an answer, it will answer the question. Otherwise, the chatbot will tell users that this question has not been included for now.

For fake tweet detection, users can enter tweet text according to the prompt, and the chatbot will return the result of AI recognition. If the AI thinks it is a fake tweet, it will also show which part is responsible for the identification of a fake tweet.

For the COVID-19 test by coughing, users can click the 'Record' button to start recording the cough audio, click 'Stop' to stop the recording, and then click 'Send' to send it to the AI for recognition. After sending, the page will display the recorded audio wave. The AI recognition time is about 30s, and the user needs to wait for a while in this interface to get the result.

5.2. UI Mockup


Bot: Hi! This is CovBot. I can assist you with anything you want to know about Covid-19. Try asking me questions about covid.

Bot: DISCLAIMER: Please note that the information provided here is not a substitute for advice from a medical professional.

Me: Is the virus transmitted by water?

Bot: The COVID-19 pandemic has had an adverse effect on human health, economy, and diverse environments. Besides the general transmission of the virus through air droplets and human-to-human contact; it is also transmitted while infecting the digestive system, which subsequently is defecated through the feces. Such fecal transmission

Me: #IndiaFightsCorona India has one of the lowest #COVID19 mortality globally with less than 2% Case Fatality Rate. As a result of supervised home isolation &amp; effective clinical treatment many States/UTs have CFR lower than the national average. https://t.co/QLiK8YPP7E

Bot: This is a real news.

Me: Alfalfa is the only cure for COVID-19.

Bot: This is a fake news. The contributions of the components in the sentence to the results are shown below.

alfa ##lf ##a is the only cure for co ##vid - 19 .

Bot: This is a fake news. The contributions of the components in the sentence to the results are shown below.

alfa ##lf ##a is the only cure for co ##vid - 19 .

Bot: Now try to record your cough audio and have a covid test. Click to start and click to stop.

Me: [audio]

Bot: Your Covid Test: Negative.

Figure 9: UI with results for all functionalities

5.3. System Architecture

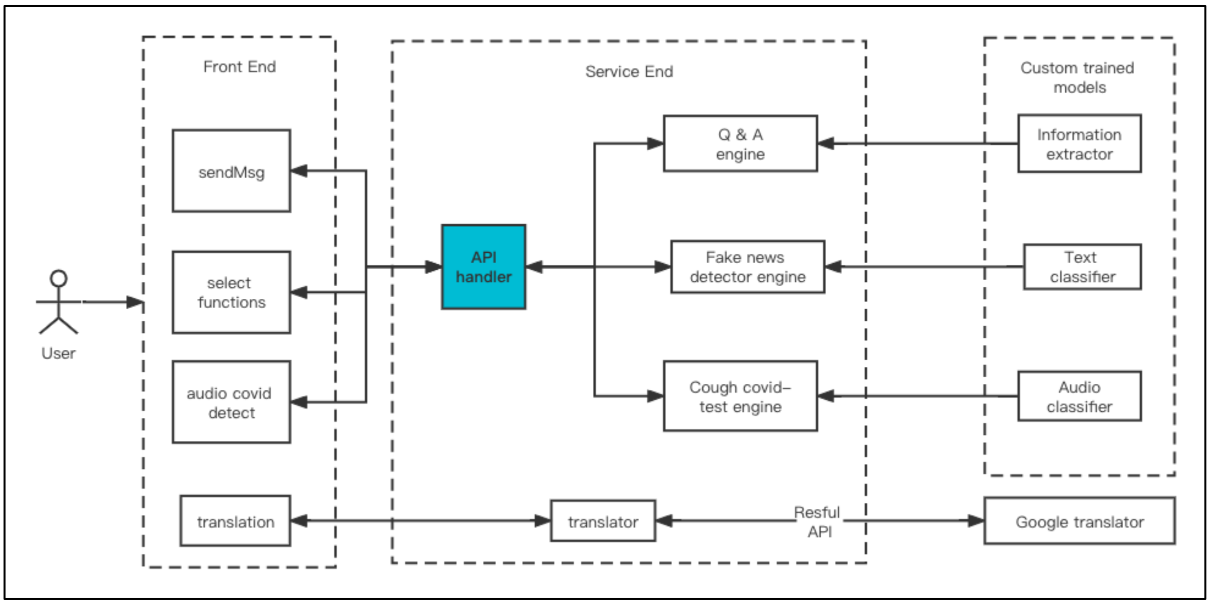


Figure 10: System Architecture

The system adopts B/S (browser/server) architecture. The front-end system implements the user interface and interactive functions of the chatbot and can interact with the server through the RESTful API. In order to be more convenient and efficient, the front end uses Vue as the framework for development.

The back-end system implements three engine classes, which deploy the three AI models separately implementing the model inference function and providing the interfaces required for external calls. Finally, the back-end system provides the RESTful API to the front-end call through the API handler. The backend system is developed in python and uses flask as a simple web framework. The overall architecture of the system is shown in Figure 10.

6. CONCLUSION AND FUTURE WORK

6.1. Conclusion

In this work, we have developed a medical domain-specific chatbot system called CovBot. The ASD Bot is capable of answering queries related to COVID-19. In addition to that, it also provides services like 'Fake tweet detection' and 'COVID-19 screening using cough audio recordings'. This report provides a detailed explanation of all these features and includes the design and implementation steps.

6.2. Future Work

Due to time and resource constraints, there are a few limitations to this chatbot.

6.2.1. FAQ Answering

For now, we have included only 38 relevant question-answer pairs for FAQ answering in the CSV file. Due to the limited amount of QA pairs available, all types of user questions may not match in the real-life scenario. By introducing more QA pairs, we plan to improve the quality of interaction between chatbot and user in the future. Secondly, we have used the extractive

QA method, which gives unnatural answers. In the future, we plan to use abstractive QA using GPT to generate natural/human-like answers.

6.2.2. Fake Tweet Detection

We currently only focus on the tweets dataset in this project. In order to handle more user scenarios, we will extend the domain to other news sources like Facebook, Instagram, newspaper, and other open-source dataset provided by research labs like COVID-19-rumor-dataset [25].

6.2.3. COVID-19 Screening Using Cough

Virufy dataset has a very low number of samples and the class distribution is imbalanced. Moreover, the data was captured over a short period. In the future, we plan to combine multiple open-source datasets available for the task such as Virufy [24], Coughvid [26], and Coswara [27] to increase the number of samples, handle class imbalance problems, and get data captured over a longer period. Moreover, in the future, we would also like to extract other statistical values besides mean and std from acoustic features. For example, 25th percentile, median, 75th percentile, variance, and mode.

6.2.4. Other Improvements

Currently, CovBot provides only two additional features apart from answering FAQ questions. In addition to this, including features like finding institutions such as hospitals and vaccination centers near the user's location, and appointment booking in those institutions will help the people to a great extent.

REFERENCES

1. https://www.sciencedirect.com/science/article/pii/S0140673620301835
2. https://apps.who.int/iris/handle/10665/331475
3. https://reliefweb.int/report/china/report-who-china-joint-mission-coronavirus-disease-2019-covid-19?gclid=Cj0KCQjwgMqSBhDCARIsAIIVN1Vq3Gp196iYjypI-TvIo6BVT_ShFCeihTbe3zkMNy59XMcTzKi-v9AaAnqhEALw_wcB
4. https://pubmed.ncbi.nlm.nih.gov/32101510/
5. https://link.springer.com/article/10.1007/s10238-020-00650-3
6. https://link.springer.com/article/10.1007/s13369-021-05810-5
7. https://www.kaggle.com/datasets/allen-institute-for-ai/CORD-19-research-challenge
8. https://www.kaggle.com/code/xhlulu/cord-19-eda-parse-json-and-generate-clean-csv
9. https://www.analyticsvidhya.com/blog/2021/05/build-your-own-nlp-based-search-engine-using-bm25/
10. https://arxiv.org/abs/1910.01108
11. https://arxiv.org/abs/1810.04805
12. https://www.sbert.net/
13. https://pesquisa.bvsalud.org/global-literature-on-novel-coronavirus-2019-ncov/resource/pt/covidwho-1320
14. https://www.kaggle.com/datasets/elvinagammed/covid19-fake-news-dataset-nlp
15. https://doi.org/10.3390/info10040150
16. https://huggingface.co/docs/transformers/model_doc/bert#transformers.BertForSequenceClassification
17. https://github.com/jessevig/bertviz
18. https://www.moh.gov.sg/covid-19/general
19. https://en.wikipedia.org/wiki/COVID-19_pandemic
20. https://pubs.acs.org/doi/10.1021/acsnano.0c02624
21. https://www.who.int/news/item/13-12-2017-world-bank-and-who-half-the-world-lacks-access-to-essential-health-services-100-million-still-pushed-into-extreme-poverty-because-of-health-expenses
22. https://www.nytimes.com/2020/06/16/upshot/coronavirus-test-cost-varies-widely.html
23. https://dergipark.org.tr/en/pub/ejosat/issue/64234/1010723
24. https://github.com/virufy/virufy-data
25. https://www.frontiersin.org/articles/10.3389/fpsyg.2021.644801/full
26. https://github.com/virufy/virufy-cdf-coughvid
27. https://github.com/iiscleap/Coswara-Data

APPENDIX

COVID-19 Screening Using Late Fusion Model

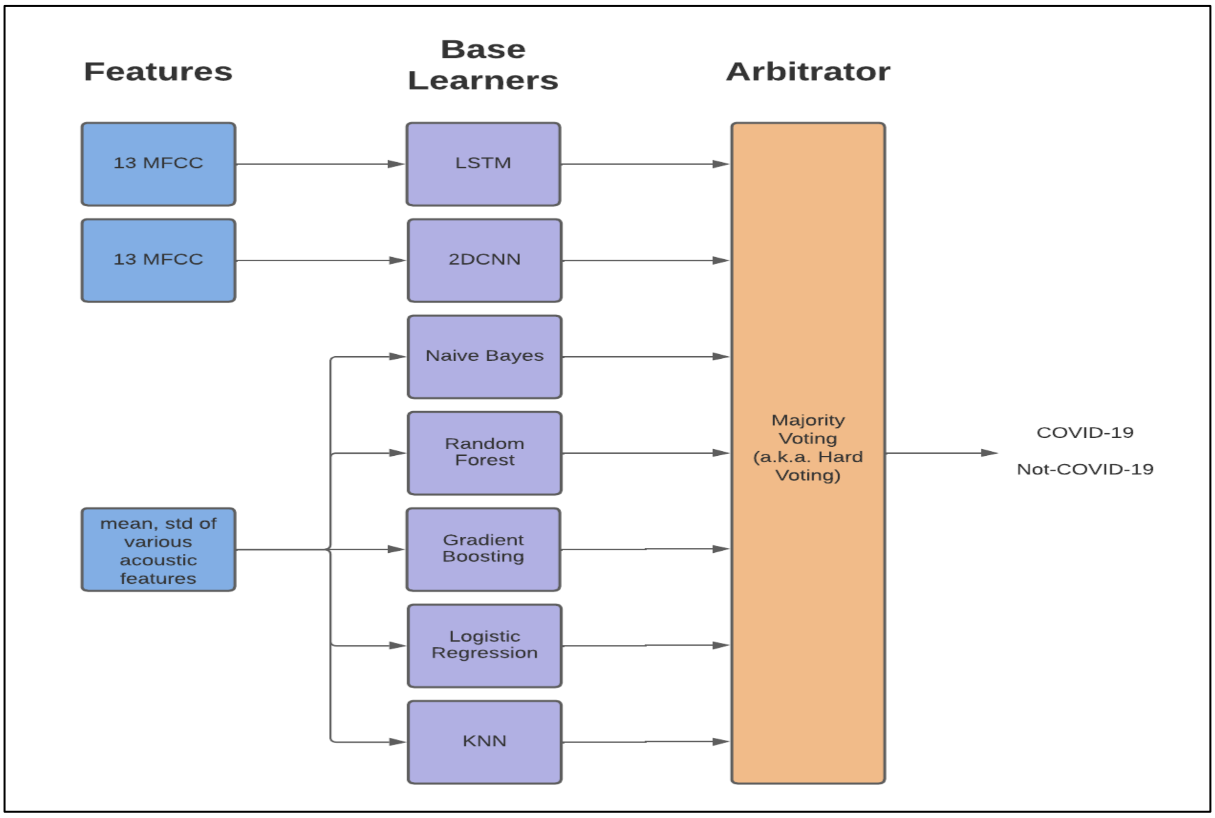


Figure 11: Proposed Late Fusion Model Architecture for COVID-19 Screening

7 models which performed best in COVID-19 screening using cough were chosen for an ensemble. All models (base learners) output COVID-19 or Non-COVID-19. The system takes in these predictions from the base learners and outputs the class which has the highest number of votes. This method is known as Majority/ Hard Voting. But a thorough performance evaluation was not done on the proposed late fusion model. The architecture of the proposed late fusion model is shown in Figure 11.

COVID-19 Screening Using DWT Feature Extraction

Table 10: Overall performance of DWT+ML models on Virufy test data

Model	**Training Accuracy**	**Testing Accuracy**	**Testing Precision**	**Testing Recall**	**Testing F1 score**
Naive Bayes	0.64	0.67	0.67	0.67	0.67
Random Forest	0.97	0.78	0.79	0.78	0.78
Gradient Boosting	1.00	0.82	0.82	0.82	0.82
Logistic Regression	0.53	0.54	0.77	0.50	0.36
KNN	0.81	0.70	0.70	0.70	0.70
SVM	0.54	0.54	0.27	0.50	0.35

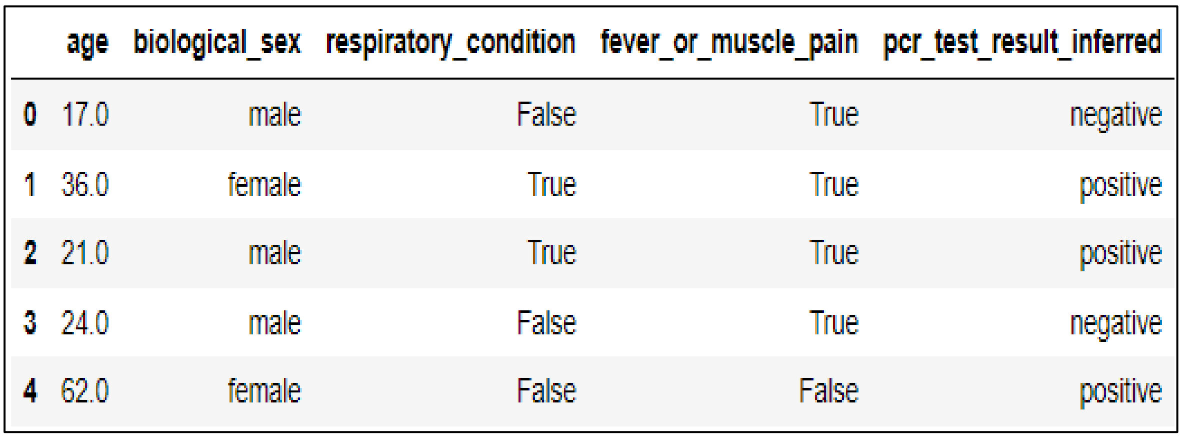
Table 11: Hyperparameters values set for extracting DWT features from cough samples

Hyperparameter	**Value**
Wavelet (Filter)	Daubechies wavelets of the fourth order (db4)
Number of decomposition levels	Maximum level of decomposition possible
Padding	Constant

Besides extracting features from the 1/10 sec audio signal/sample using Fourier Transform (FT), we extracted features (approximation and detail coefficient lists) using Discrete Wavelet Transform (DWT) as well. 25th percentile, median, 75th percentile, mean, standard deviation (std), variance, zero crossing rate, and shannon entropy value (a measure of complexity of the signal) were calculated from approximation and detail coefficient lists and used as final features for modeling.

The performance of ML models trained on DWT features is shown in Table 10. As evident from the table, the models are unable to perform/generalize well on DWT features. Choosing the best set of DWT hyperparameters (wavelet type, number of decomposition levels, padding, etc.) is a very time and resource consuming process. In this project, we tried training models with only one randomly chosen set of DWT hyperparameters which are shown in Table 11.

Covid Screening Using Patient-Level Contextual Information



	age	biological_sex	respiratory_condition	fever_or_muscle_pain	pcr_test_result_inferred
0	17.0	male	False	True	negative
1	36.0	female	True	True	positive
2	21.0	male	True	True	positive
3	24.0	male	False	True	negative
4	62.0	female	False	False	positive

Figure 12: First 5 rows of pre-processed Coughvid meta-data

Table 12: Overall performance on Coughvid test meta-data

Model	Training Accuracy	Testing Accuracy	Testing Precision	Testing Recall	Testing F1 score
Naive Bayes	0.58	0.52	0.57	0.57	0.52
Random Forest	0.66	0.46	0.46	0.45	0.45
Gradient Boosting	0.66	0.43	0.45	0.45	0.43
Logistic Regression	0.58	0.53	0.55	0.56	0.53
KNN	0.63	0.52	0.53	0.54	0.52
SVM	0.53	0.43	0.69	0.57	0.35

To check if there is a strong correlation between patient-level contextual information and cough classes (COVID-19/Non-COVID-19), we decided to develop a classification model trained on patient-level contextual information.

Coughvid dataset was used for this analysis. The dataset consists of 22,040 records collected from different patients. There are 15 columns in the metadata. Out of those 15 columns only 4 provided useful patient-level contextual information for diagnosis – age, biological sex, respiratory condition, fever, or muscle pain. Hence these were selected for model training. A series of pre-processing steps were also applied to clean the data. So final dataset had 690 rows and 5 columns. Figure 12 shows the first 5 rows of the pre-processed dataset.

The performance of ML models trained on patient-level contextual features is shown in Table 12. From the results, we observe that model is not performing much better than a random classifier. But the results are inconclusive because of limited data available.