Reply to critique review:

G3:

- What is your expected result of your proposed model? How could your results be quantitatively measured or compared to other state-of-the-art models like LSTM, CNN, and so on?
  This is included in the results section in this report.
- From your slides, I saw only one data from cough, I guess this is probably from negative cough data. For your observations, since MFCC demonstrates clearer distinction with different color denotation, how did you tell if one data is positive or negative based on the color denotation?

  We are not training based on the color but more MFCC features, the colored spectrum is just a way of better demonstrating.
- When processing data, the first step is to delete some useless data, based on which rules that you can make sure that some data is useless and why?
  By reading the recording files length. The file length of useless data is different from the normal ones.

G34:

- The design only used one feature (MFCC), which might be insufficient since there exist a lot of features that could be extracted from an audio file such as zero crossing rate, spectral centroid and spectral roll off etc. These features may not be useful to the classification, but worth trying.
  Thanks for your advices and we have test through several features, among which the MFCC still runs best.
- In the model design part, the speaker did not mention why they would choose to use 2 of each same conv2d layer (maybe typo?). The straight fix for this is just to delete the duplicated layers.
- No, we meant to create deeper network with better result.
- As I mentioned in comments, there was no result provided in both presentation’s ppt and the code running part, this was not a good style of presentation. It only showed the model summary, which is insufficient at all. Please at least add or state what you are currently focusing or current process.
- Now we have the results.

G71:

- How was the performance of the model that was run on the mobile application?

  We have no mobile application.
- 1287 negative cough samples, and 89 positive cough samples. How was the class imbalance addressed?
- The results of the algorithm were not discussed in the presentation? How did your model perform with the given dataset? What performance metrics were used to test the validity of the model?
- It was mentioned this method was cheap and easy for people to use, do your team foresee this as something doctors will adapt in a clinical setting? If so, what ensures the trustworthiness of the results?

Group54: Preliminary COVID-19 Diagnosis Based on Cough Recording Classification

Jiayu Zhao[#1], Zhuoran Liu[*2]

[#]*University of California, San Diego*

[1]jiz071@eng.ucsd.edu

[2]zhl003@eng.ucsd.edu

I. Introduction

In the light of the COVID-19 pandemic, people all around the world are suffering from pain and fear, tons of people lost their family members and are facing unemployment crisis due to business shutdown. By June 13, COVID-19 had 7,751,747 confirmed cases and 429,062 global deaths. [1] Given the fact that no vaccine is developed to combat this disease as of now, the only way to minimize the spread of COVID-19 is timely detection and isolation if tested positive. However, almost all of the tests using right now must be conducted on site, which not only draw a lot of pressure on the medical system but also miss the best treatment time for the infected people. Moreover, it may cost a lot of money to do the test for people who are not covered by medical insurance. Obviously, a scalable, accessible, costless and effective diagnosis method needed to be developed for the preliminary test of COVID-19 to address current issues.

In this project, we proposed to use cough as the preliminary diagnosis for COVID-19. To this end, we took the positive and negative cough audio files as input, and we then used two deep learning algorithms: Convolutional Neural Network(CNN) and Long Short Term Memory(LSTM) to output a predicted positive or negative test result. The performance of these two different algorithms were compared and discussed. We also explore different parameters' effect on the performance of our algorithm.

II. Related works

The existing works have proved that coughs of different respiratory diseases have distinct latent features.[2] By appropriate extracting those features from the audio files, it is possible to train a sophisticated deep neural network(DNN) for cough detection and classification. Liu et al demonstrated a two step cough detection algorithm using DNN and hidden markov model(HMM).[3] It turns out that the DNN based method outperform the traditional Gaussian Mixture Model(GMM) in terms of sensitivity, specificity and F1 measure respectively. This is mainly because DNNs can combine different features in an easier way and its complex structure allows it to learn from a large amount of data. More recently, Amoh et al reported a cough detection method using Convolutional Neural Network(CNN) and Recurrent Neural Network(RNN).[4] They preprocessing their data by converting the audio data into spectro-temporal 'image' using Short-Time Fourier Transform(STFT). Given the fact that images for CNN have been thoroughly studied, and there are a lot of software resources available, CNN can be easily adopted for cough detection in this case. On the other hand, RNN is a class of nets that can analyze time series data. With specialized cells like LSTM and Gated Recurrent Unit(GRU), one is able to train an RNN on long sequences, which would benefit the cough detection task. In conclusion they claim that CNN yields a better specificity whereas the RNN produces the better sensitivity.

A series of researches in COVID-19 detection and diagnosis throw light on our work. Coswara[5], a dataset that collected from user applications used a detection function that includes features of 28 dimensions. The MFCC scale is the feature that has the highest dimension, i.e. 13 dimensions. This made us believe that MFCC may be a good feature in demonstrating the cough recordings. However, the model is more suitable

for detection whether a recording contains any cough sound, and this step is not applicable in our project design. Thus, there are still some problems to overcome.
Also, an earlier work named Flusence[6] drew our attention. The system provides contactless influenza detection, but after careful analysis, we found that the system needs more than recording. For example the thermal cameras will provide biological data of the patients. Even though the detection model is not realistic in our design, the CNN-based recognition algorithm is advisable. The model consists of 2 convolution layers with size of 98*128, 3 convolution layers with size of 47*64, and three layers of 25*32. In spite of the different target numbers of the classifier, we still found this is a satisfying design.

III. Dataset and Features

A. The COVID-19 Dataset and preprocessing

In this project, we used the COVID-19 data set provided by TAs, which contains 1287 negative cough audios and 89 positive cough audios. One of the challenges of this project is that these cough audios have different lengths in time, meaning that they can not directly feed into the CNN model, as it requires input data have a known and fixed dimension. To solve this problem, we use a zero-padding method to fix the dimension of input.

B. Feature extraction

We then convert our cough audios of two classes into Mel scale and perform Cepstral analysis on the Mel spectrum to compute their Cepstral coefficients, which are also known as Mel-frequency cepstral coefficients (MFCCs). It provides a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. The implementation of MFCC in this project relies on a python library named librosa. Several steps are needed in order to generate the result. Frame blocking provides more stable data slices, windowing with Hamming window can filter the data that is unwanted, FFT extract the spectrum in a concise way, and Mel-Scale help the spectrum transformation by this equation:

$$f_{mel} = 2595 * log_{10}(1 + \frac{f}{700}) \qquad (1)$$

The figure???? clearly demonstrate the transformation between frequency and Mel-scale frequency. Mel-scale is a good representation of the human hearing domain, which is beneficial for human voices processing.

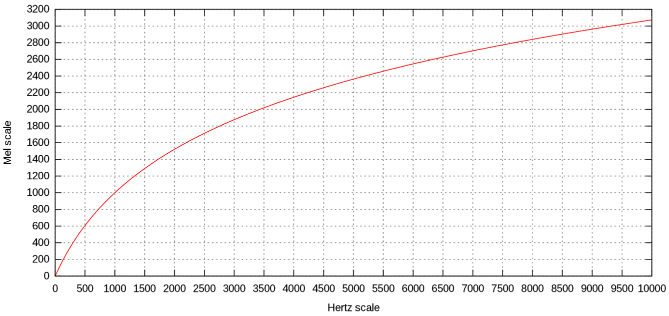


Figure 1: Frequency to Mel-frequency curve[10]

The process of MFCC is concluded in the following block diagram, which represents the data processing in the project. The output data in the project is MFCC feature in 2 dimensions, that is 11*40, 11 represents the maximum padding size, and 40 is the MFCC feature number.
Combining with the positive or negative label of the data, the recordings transformed into 1349 pieces of MFCC matrix. The data is stored in a json file named “data.json” for later data processing. We separated 25% data--338 pieces-- as testing data, and we divided the remaining data into two parts--80% as training dataset, which takes 808 pieces, and 20% as validation dataset, which avoided the overfitting during the model training.

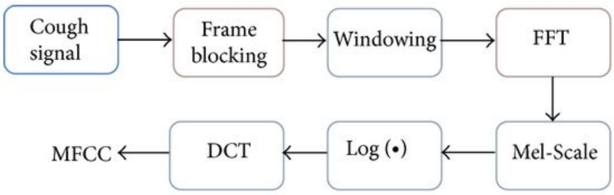


Figure 2: Block diagram of MFCC processing[9]

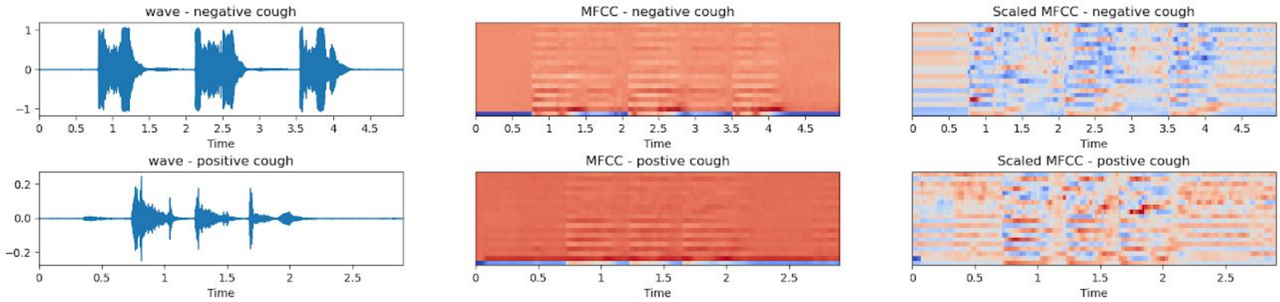


Figure 3 : Examples of Waveform and MFCC features

IV. Methods

A.Cross validation and Confusion matrix scenario

A series of parameters could be modified to find the best combination of the model. For the data preprocessing, we have multiple choices for the data padding because the recordings are ranging from 2 seconds to 11 seconds. We test different maximum padding sizes including 2,3,5,6,11. As for the n_mfcc parameter in the mfcc function, we test the features with a number of 13 and 40.

As for the training epochs and training batch size, we tested training epochs equals to 30, 50, 70 and 100. For the batch size, we tested values ranging from 32 to 80.

The confusion matrix was designed to demonstrate the precision of the model and by observing the false positive and false negative values, we can identify whether the system suffered from overfitting.

B.Convolutional Neural Network(CNN)

The CNN model was used to create a binary classifier. The shape of the input training dataset is (808,40,11,1). It means that the dataset consists of 808 pieces of data that has 40 features and 11 padding size, with 1 label denoting positive or negative.

The convolution layers we used are two dimensional convolution layers: conv2d().

The first two convolution layers are in a size of (64, (2, 2), padding='same', activation = "relu"), and the third and the fourth layers are designed as (32, (2, 2), padding='same', activation = "relu"). We held the belief that deeper layers will bring better results and it behaves as we expected. Then the pooling layers cut down the dimension of the data and speed up the calculation by using two Maxpooling layers whose shape are: ((3, 3), strides=(2, 2), padding='same'). Each Maxpooling layer is attached after the convolution layers with the same strides except the last convolution layer. The last convolution layer is followed by a Flatten layer, which presses the data into three dense layers computing the final classification results. The dense layers press the data size from 256 to 64, and finally reach 2 categories. In order to prevent overfitting, we add a dropout layer with a dropout rate = 0.5 at the last two dense layers. The number of total parameters are 47,265, which consists of 47,009 trainable parameters and 256 non-trainable parameters.

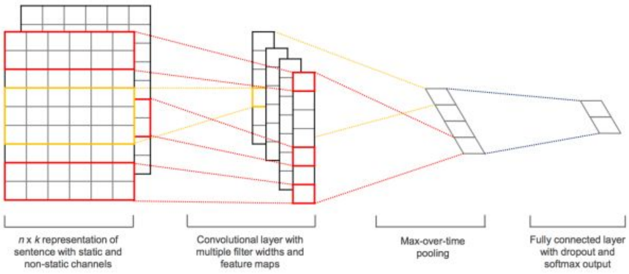

n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 4: CNN model for binary classifier[8]

C. Long Short Term Memory(LSTM)

As mentioned earlier in section Ⅱ, the problem of CNN is that the input dimension must be known priorly and fixed, thus it can not handle data variance in length, such as time series data. LSTM is proved to be robust on detecting long-term dependencies in the data.

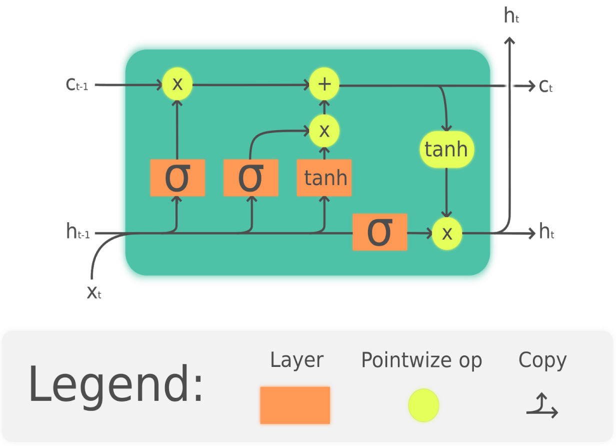


Figure 5: LSTM cell[10]

As shown in the **Fig 5**, there are four layers in the LSTM cell, in which the "tanh" layer is the main layer and the three other sigmoid layers are gate controllers. The first gate controls what information of the long-term state should be thrown away, the second gate controls what information should be stored in the long term state and finally, the third gate which parts of the long term state should be extracted. Basically, an LSTM cell can learn to recognize an important input, store it in the long term state and extract it whenever it is needed. After several tried, we choose LSTM as the first layer, followed by two fully connected layers with activation functions of "relu" and "softmax" , respectively.

V. Experiments and results

Data Processing

The combination we tested are n_mfcc and maximum padding size, the n_mfcc choices are 13 and 40, which are both frequently used settings. The maximum padding size is in the set of 2,3,5,6,11. By testing the 10 combinations, we found an increasing accuracy ranging from 95.4% to 97.6%, indicating that the more features were kept, the better result it generated. We also double checked that the zero padding will not influence the shape of the wave in frequency domain, and we make sure that zero padding will not influence the majority of the data, which mainly take 2 to 3 seconds. But the larger size and more features also have disadvantages, that is this will take longer time to extract the feature. Because longer pieces means more data point to sample.

The best combination we find is 11 zero padding size and 40 MFCC features.

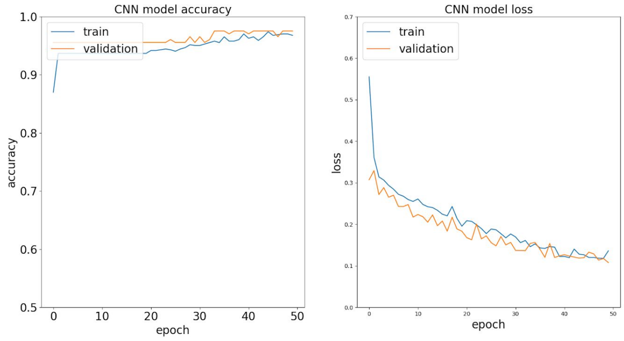
Training parameters selection

The parameters that can be trained and modified are training epochs and training batch size. If the epochs and batch size are too small, the model may suffer from under-fitting. But with too many epochs and batch size, the model is burdened and results in overfitting. The balance between accuracy and scalability should be tested. Finally, we got the combination of 50 epochs training with batch size equals to 70.

Model selection

The model we choose is CNN and LSTM. The metric loss function is binary_crossentropy, since we are going to generate a binary classifier. Comparing the best result of each, we found that LSTM behaves better than CNN model. Detailed information is in the following part.

Results



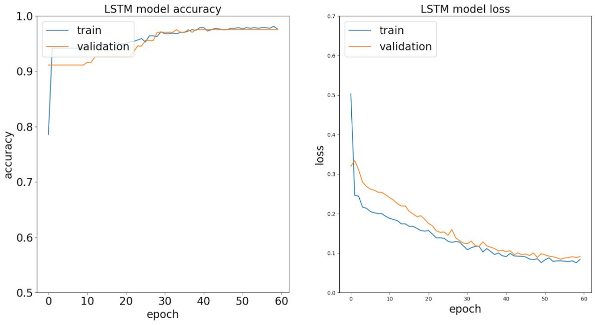
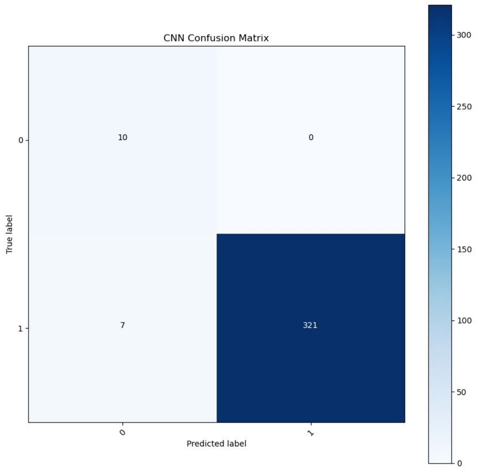


Figure 6: Accuracy and loss of CNN (upper) and LSTM(below) mode

1

After 50 epochs training with 70 batch size, the accuracy result of CNN is accuracy result of CNN is 0.9763 and the LSTM accuracy is 0.9792. (**Fig 6**)



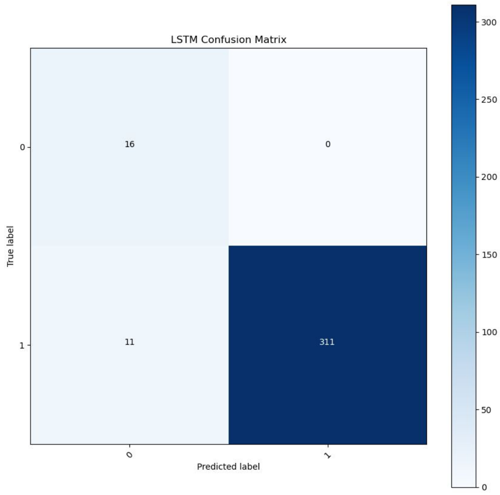


Figure 7: Confusion matrix of CNN(upper) and LSTM(below) model

Comparing the confusion matrix of the CNN model and LSTM model(**Fig 7**), we found that even though the LSTM data has relatively large false positives data, it has zero false negatives data. Since this is a recognition in the medical domain, we held the belief that rather regards more negative patients as positive than ignoring some of the patients as negative. This will make sure that no patient is ignored. This is more important than seemingly fancy results.

VI. Conclusion

In this project, we explored two types of models and implemented them on the MFCC features extracted from the COVID-19 cough recording data. The best part of the project was we are not satisfied with the existing results when the CNN model reached 95% accuracy and kept moving forward to better implementation with the LSTM model. The result we showed in the previous part indicates that we not only get an accurate prediction, but also the ability to generalize the model to more dataset.

Since time is limited and the group only has two members, some ideas have no choices to implement at once. We still wish to form a more comprehensive feature extracting process that combined energy and other features of the sound data. Also, if time allows, we want to implement some powerful models such as VGG-16 etc. This project is worth more analysis in the future and we are looking forward to deeper research in it.

REFERENCES

[1] COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU). Accessed on: June 13, 2020. [Online]. Available: https://coronavirus.jhu.edu/map.html

[2] C. Bales, M. Nabeel, C. N. John, U. Masood, H. N. Qureshi, H.Farooq,I. Posokhova, and A. Imran, “Can Machine Learning Be Used to Recognize and Diagnose Coughs?” arXiv preprint arXiv:2004.01495, 2020.

[3] Liu, Jia-Ming, et al. "Cough detection using deep neural networks." *2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2014.

[4] Min Xu; et al. (2004). "HMM-based audio keyword generation" (PDF). In Kiyoharu Aizawa; Yuichi Nakamura; Shin'ichi Satoh (eds.). Advances in Multimedia Information Processing – PCM 2004: 5th Pacific Rim Conference on Multimedia. Springer. ISBN 978-3-540-23985-7. Archived from the original (PDF) on 2007-05-10.

[5] Sharma N, Krishnan P, Kumar R, et al. Coswara--A Database of Breathing, Cough, and Voice Sounds for COVID-19 Diagnosis[J]. arXiv preprint arXiv:2005.10548, 2020.

[6] Al Hossain F, Lover A A, Corey G A, et al. Flusense: A contactless syndromic surveillance platform for influenza-like illness in hospital waiting areas[J]. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2020, 4(1): 1-28.

[7] J.Brownlee. “Best Practices for Text Classification with Deep Learning?”[On-line] Retrieved on June 5th 2020. from

https://machinelearningmastery.com/best-practices-document-classification-deep-learning/

[8] Al-Momani O, Gharaibeh K M. Effect of wireless channels on detection and classification of asthma attacks in wireless remote health monitoring systems[J]. International journal of telemedicine and applications, 2014, 2014.

[9] https://en.wikipedia.org/wiki/Mel_scale

[10]https://en.wikipedia.org/wiki/Long_short-term_memory

Contributions

Jiayu is working on the data preprocessing and LSTM model

Zhuoran is working on the feature extraction and CNN model