Towards Mitigating Spreading of Coronavirus via Mobile Devices

Shashank Reddy Danda
Department of Computer Science
Michigan Technological University
Houghton, MI, USA
sdanda@mtu.edu

Bo Chen*
Department of Computer Science
Michigan Technological University
Houghton, MI, USA
bchen@mtu.edu

***Abstract*—Recently, impact of coronavirus has been witnessed by almost every country around the world. To mitigate spreading of coronavirus, a fundamental strategy would be reducing chance of healthy people from being exposed to it. Having observed the fact that most viruses come from coughing/sneezing/runny nose of infected people, in this work we propose to detect such symptom events via mobile devices (e.g., smartphones, smart watches, and other IoT devices) possessed by most people in modern world and, to instantly broadcast locations where the symptoms have been observed to other people. This would be able to significantly reduce risk that healthy people get exposed to the viruses. The mobile devices today are usually equipped with various sensors including microphone, accelerometer, and GPS, as well as network connection (4G, LTE, Wi-Fi), which makes our proposal feasible. Further experimental evaluation shows that coronavirus-like symptoms (coughing/sneezing/runny nose) can be detected with an accuracy around 90%; in addition, the dry cough (more likely happening to COVID-19 patients) and wet cough can also be differentiated with a high accuracy.**

***Index Terms*—COVID-19 symptoms, Mobile devices, Feature extraction, Detection, Location**

I. Introduction

To prevent coronavirus from spreading, the best strategy would be requiring a person to remain in quarantine once infected. This is unfortunately not realistic, because: Symptoms of coronavirus are difficult to be differentiated from those of cold and flu, and most of infected people may still wander around before they are actually confirmed through coronavirus testing, which usually suffers from a long delay. Another strategy would be requiring any person who has flu/cold/coronavirus symptoms to stay at home and avoids going to public places. This may not be effective either, because: First, a potentially infected person may live alone, and he/she sometimes really needs to go outside even if he/she has symptoms, e.g., to purchase household necessities, to see a doctor. Second, some of the infected persons may not have obvious symptoms [2].

Social distancing could definitely help, but may not solve all the issues. According to [1], "The new coronavirus is a respiratory virus which spreads primarily through droplets generated when an infected person coughs or sneezes, or through droplets of saliva or discharge from the nose". What if an infected person just coughed or sneezed in a public location, and left. The droplets may remain in the air or viruses may remain in the stuffs around (e.g., a self-checkout machine in Walmart). Another person who just passes through may be exposed to the viruses. This issue clearly cannot be solved by social distancing.

The problem being faced. To mitigate coronavirus spreading, a fundamental strategy would be reducing chance of healthy people from being exposed to the coronavirus. Having observed the fact that most viruses come from coughing/sneezing/runny nose of the infected people, if we can come up with a way to capture and publish each coughing/sneezing/runny nose event, we should be able to significantly reduce risk that healthy people get exposed to the viruses. For example, a healthy person will choose to stay away from a location/trajectory where another person just coughed half a minute ago in a closed space like a supermarket; a store staff will put more efforts on cleaning the locations where coughing/sneezing/runny nose happen (while wearing a face-mask for self-protection). However, capturing and publishing every coughing/sneezing/runny nose event is indeed a challenging problem in real world.

Proposed approach. Thanks to the broad use of mobile devices, this problem turns to be solvable. Nowadays, almost every person has a smart device (e.g., a smartphone, a smart watch), and each smart device is usually equipped with various sensors including microphone, accelerometer, GPS as well as network connection (4G, LTE, WiFi). The smart device can be utilized to 1) detect a potential coughing/sneezing/runny nose event; and 2) publish this event as well as its GPS location via Internet. Note that the proposed approach will be useful only after more and more users participate the network (e.g., using their own mobile devices to capture their own symptoms or symptoms happening around and distributing them to the network). The participants will have motivation since they will also benefit from obtaining symptoms events from other participants to avoid trajectories/places where chance of being exposed to coronavirus is high; in addition, each event will not be associated with the owner of the device, protecting the

This is a preprint of the paper "Towards Mitigating Spreading of Coronavirus via Mobile Devices", which will appear in IEEE Internet of Things Magazine. The copyright has been transferred to IEEE, and a link to the article abstract in IEEE Xplore will be added when available.

Bo Chen is the corresponding author. Email: bchen@mtu.edu.

owner's privacy.

In this paper, we focus on assessing the feasibility of using mobile devices to detect potential symptoms including coughing/sneezing/runny nose events (Sec. II). We extract a few useful features from those symptoms for detection, and our experimental results using real-world data sets show that we can detect those symptoms with a high accuracy, i.e., more than 90% as shown in Sec. IV-A. In addition, having observed COVID-19 patients more likely suffer from dry cough rather than wet cough, we therefore also assess the feasibility of differentiating dry and wet cough using mobile devices (Sec. II-B). Our experimental results show that we can differentiate the two types of cough effectively, i.e., more than 90% detection rate as shown in Sec. IV-B. We also discuss a few technical details during implementing the detection approaches into Android devices (Sec. III).

II. Detecting Coronavirus-like Symptoms

A. *Detecting Cough/Sneeze/Runny Nose*

The coronavirus-like symptoms like coughing, sneezing, or runny nose will produce audio signals, and we focus on detecting such symptoms by analyzing the audio signals. The process for detection is described in Fig. 1. Given the audio signal (captured from the sensor equipped with a mobile device), we first extract a few time/frequency domain features, and perform feature normalization using the standard scaler (note the normalization can help reduce the time needed for calculation in the model). Once the aforementioned pre-processing is done, we can perform classification using the machine learning (ML) model which has been trained using a training data set. In the following, we elaborate both the features and the classification models we use in the detection.

Features. We need to convert the audio signals to some representation which usually consists of several features that describe each frame of the audio signal. The common features used in speech recognition systems are:

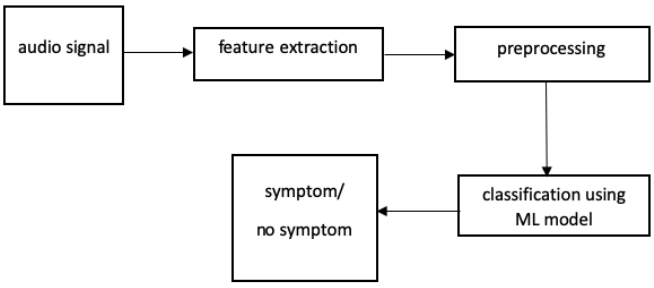

Fig. 1. The process of detecting coronavirus-like symptoms.

- **Mel-frequency cepstral coefficients (MFCC):** MFCC is one of the most broadly used features for speech classification [13]. MFCC makes learning of an audio signal easy and the total MFC coefficients range from 0 to 39 which makes a complete power spectrum of a signal on a Mel scale frequency. It considers the human perception for sensitivity mostly by transforming the frequencies to Mel Scale. We consider a total of 20 coefficients to achieve a good performance. The frequencies are converted to Mel scale using the equations mentioned in [13].
- **Variation rate:** It measures the variation coefficient of the short term energy [13] which is calculated by adding the squared absolute values of amplitudes normalized by the length of the frame.
- **Zero crossing rate (ZCR):** ZCR measures the sign changes of the amplitude values of each frame [13]. We count the number of zero crossings which helps to discern no symptom sounds by setting up a threshold.
- **Entropy:** Probability of short-term energy for each frame after dividing it into several sub frames is called entropy. The entropy can be used to show variation between the sounds with symptoms and without symptoms.
- **RMS:** RMS measures the energy for each frame and mostly helps in differentiating the frames with higher and lower energies. With the help of RMS we can separate the audio signals because the sounds with symptoms will be having a higher RMS [14] when compared to the other signals.
- **Spectral bandwidth:** Bandwidth is mostly useful in measuring the uniformity of the FFT spectrum [14].
- **Spectral centroid:** Characterization of the spectrum can be done using the spectral centroid [14]. It measures the centroid of the spectrum and gives the total number of frame count present in a particular audio sample.
- **Spectral flux:** Changes in the spectral energy between two successive frames is measured using the spectral flux. It compares the power spectrum of both the frames and shows the variation among those frames.
- **Chroma features:** These features venture the sound events onto 12 bins which consists of 12 pitch classes. Most of the harmonic and the melodic features can be found easily by these chroma features.

Classification models. Support vector machine (SVM) and random forests (RF) are widely used for large training data sets which usually take less computational time when compared to other algorithms like kNN. Especially, SVM has been used in most of the research related to cough and other symptoms detection [14]. We therefore perform two types of classification based on the SVM and the RF, respectively: 1) a *binary classification* in which we individually detect each symptom (cough, sneeze, and runny nose) using SVM and RF; and 2) a *multi-class classification* in which all the three symptoms are detected together using a better model selected based on the results of binary classification using SVM and RF. Steps for selecting the better model in the multi-class classification are described below:

1) We try different combinations of features, and find out those features which can result in a better detection performance in both binary and multi-class classification.
2) Detection of each symptom individually is performed

using the binary classification with SVM and RF, and the model with better performance is chosen using cross validation, which acts as the final model for the multi-class classification.
3) A final data set is chosen with equally balanced classes using some data sampling methods in order to avoid class imbalance problem in the multi-class classification [14].

B. Differentiating Dry and Wet Cough

Differentiating dry and wet cough is important since COVID-19 patients more likely suffer from dry rather than wet cough. In other words, if the detected event is a dry cough, the risk of COVID-19 exposure will be much higher than that of a wet cough, and the people who receive such an event would be more alert. In this following, we describe features used for dry/wet cough classification as well as the classification models.

Features. MFCC and ZCR can be used to differentiate dry and wet cough, because: MFC coefficients are used in representing the short term power spectrum of a signal, and the power spectrum shows the coefficients for each frequency which varies in dry and wet cough signal. ZCR is used in finding the number of zero crossings in a signal, which is usually high for high frequency signals like wet cough signal due to presence of mucus and, is usually low for dry cough signal. Other features for this purpose are listed as follows:

- **Formant frequencies:** In general, the resonance of the human voice tract is known as formants. Several formants can be extracted from an audio signal but usually the first three formants carry most of the useful information since these acoustic sounds are mostly related to the upper airways. The first three formant frequencies in linear predictive coding spectrum are usually denoted as F1, F2, F3, which can be calculated using the Burg algorithm [7].
- **Log energy:** Difference between segments can be shown using the log energy which shows the variation of speech over time. This energy is calculated using the MFB output which is based on sub band and captures the dynamic variations of speech signals.
- **Pitch:** Pitch is the fundamental frequency of the signal that varies both in dry and wet cough due to presence of mucus sounds in wet cough. As per the analysis, the average fundamental frequency for the spontaneous cough sounds ranges between 350Hz and 450Hz and, for voluntary coughs, it ranges between 200Hz and 350Hz or above 450Hz. Based on this range, the signal gets divided into several bins and, after calculating the magnitude for each bin, we obtain some frequency which is known as pitch [15].
- **Bi-spectrum score:** Deep study of cough signals is necessary to classify them correctly and represent the content of frequency level. This is obtained using the bispectrum score which is the third order spectrum of a signal and preserves the Fourier phase information [15].
- **Kurtosis:** This is one of the measures for Non Gaussianity and useful for measuring the peakedness of a signal which differs in both dry and wet cough signal.

Classification models. To differentiate dry and wet cough, besides SVM and RF, we also use an additional model, Logistic Regression (LR). As a generalized linear model, the LR model uses several predictors which are independent to estimate the probability of a dependent variable, e.g., whether a symptom event is a wet or a dry cough. The dependent variable is assumed to be 1 for the wet cough and 0 for the dry cough. Using this model, we can estimate the probability of each given event and use a certain threshold (e.g., 0.5) to classify this event as a dry or a wet cough. If the probability of this event is greater than the threshold, it is classified as a wet cough; otherwise, it is a dry cough.

III. Implementation Issues in Mobile Devices

The next question is how to implement the proposed approach using real-world mobile devices. As Android is the most popular OS for mobile devices, we focus on Android devices. Machine learning (ML) in mobile computing usually occurs in two different ways: 1) ML on server side using Pytorch or TensorFlow; and 2) ML using on-device inference [9], which performs detection locally on the mobile device instead of outsourcing it to the remote server. In general, the server-based inference is more suitable for huge models requiring expensive computation; on the contrary, the on-device inference is more effective for small size models which is more suitable for our scenario providing low latency, less cost and more privacy [8]. We therefore use the on-device inference.

Capturing the audio input in Android devices. The audio sounds can be captured using the microphone equipped with an Android device. Media recorder API in Android is useful for capturing the audio sounds using the microphone with the user's permission.

Converting and loading the model. We use TensorFlow Lite, an open source deep learning framework for on-device inference. The TensorFlow Lite converter is used to convert a trained model into .tflite, which will be loaded into the memory, i.e., the assets folder of Android. The size of the models that we have converted and loaded ranges from 2.5MB to 4MB, which is much less than other ImageNet models [9]. Correspondingly, the time needed to predict the results ranges from 1s to 2s, which is pretty fast in the Android device.

Running inferences. After having loaded a model, features extracted from an audio sound will be input to the model. For detection, we rely on TensorFlow Lite inference, the process of executing a TensorFlow Lite model on-device in order to make predictions based on input data [4]. Note that an inference must be run through a TensorFlow Lite Interpreter. With the help of the built interpreter, input tensors are allocated with the required shape to detect a symptom event. If the detected event is a cough, it will be further classified as dry or wet cough. Therefore, we need to create instances of different interpreters

linked to each model so that we can execute one model after the other in the Android device.

Distributing the detected events to the network. After detecting a potential symptom event locally, the Android device will send the detected event along with the location and the timestamp to the server (e.g., a server in a public cloud provider like Amazon AWS) via Internet. Permissions such as access coarse location and access fine location should be given to the app for obtaining the GPS location data. A new service should be created in the app to send a record $< event,\ GPS\ location,\ timestamp >$ to the server, which will then broadcast this record to other participants in the network.

IV. Experimental Results

This section assesses the feasibility of detecting cough/sneeze/runny nose symptoms using real world data. We have collected audio samples for cough/sneeze/runny nose from several sources including OSF (Open Science Framework) and websites offering free relevant sounds, generating our own data sets[1]. We were not able to collect a lot of samples for runny nose. For each experiment, we divided the corresponding data set into a training set (used for training the corresponding model) and a testing set (used for testing the effectiveness of the detection). We extracted most of the spectral features using librosa library [3] for audio processing in Python using Google colab (a cloud-based Jupyter Notebook environment). Model training and testing were performed using Scikit-learn and TensorFlow libraries in Python. All the classification experiments were conducted on a PC[2] with Intel core i7 processor, 16GB memory and Windows 10 OS.

A. *Detecting Cough/Sneeze/Runny Nose*

Binary classification. We performed the binary classification using 1,719 cough/no symptom samples, out of which 1,000 samples were used for training and remaining 719 samples were used for testing. Similarly, sneeze data set consists of 1,108 samples of sneeze/no symptom, out of which 705 samples were used for training and the remaining 403 samples were used for testing. For runny nose detection, we used 551 runny nose/no symptom samples for training, and 264 samples for testing.

Multi-class classification (final model). With the best model obtained after having performed the binary classification for each symptom event, we performed the multi-class classification, in which we used totally 1,748 samples which include 661 cough samples, 431 sneeze samples, 150 runny nose samples, and 506 no symptoms audio samples for training. In the multi-class classification, we reduced the samples in cough and sneeze, to prevent the class imbalance problem in which the model may have a low predictive accuracy for the infrequent classes. After having trained the model, we performed testing using 1,001 mixed samples.

1) Classification Results for The Binary Classification: The features we used include MFCC, Variation rate, ZCR, Entropy, RMS, spectral bandwidth, spectral centroid, spectral flux and chroma. They were shown to be able to achieve a good trade-off between the detection performance and the computational cost in our experiments. The experimental results are shown in Table I. We can observe that: 1) For *cough detection*, we have achieved a better detection rate of 94-96% using the RF model with the number of estimators 200. The true positive rate (TPR) is 94% and the overall positive predictive value (PPV) and F1 score are both 94%. 2) Similarly for *sneeze detection*, the detection rate is better using the RF model which is 92-93%. The overall TPR, PPV and the F1 score are all 92%. 3) For *runny nose detection*, we achieve a better detection rate (around 95-96%) using the RF model, and the TPR, PPV, F1 score are all 95%.

TABLE I
Binary classification of each symptom event

Type of Event	Model Type	Performance metrics			
		Accuracy	*TPR*	*PPV*	*F1 score*
Cough	RF	94-96%	94%	94%	94%
	SVM	87-89%	88%	87%	87%
Sneeze	RF	92-93%	92%	92%	92%
	SVM	80-81%	80%	80%	80%
Runny nose	RF	95-96%	95%	95%	95%
	SVM	90-92%	90%	90%	90%

2) Classification Results for The Multi-class Classification: Based on the results obtained in the binary classification, we chose the RF model as the final model for the multi-class classification. We used the same set of features as those used in the binary classification. The overall accuracy for the multi-class classification is around 92-94% using the RF model, computed from the total number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) obtained in the experiment, and the results for each individual event are shown in Table II. We can observe that: 1) The runny nose detection has the best TPR, PPV and F1 score among all the events due to the least false positives and false negatives observed in the experiment; 2) The sneeze detection has the worst TPR, PPV and F1 score when compared to the other events because of the most false positives and false negatives observed in the experiment.

TABLE II
Multi-class classification of all the symptom events

Type of Event	Performance metrics		
	TPR	*PPV*	*F1 score*
cough	91%	93%	92%
sneeze	86%	89%	87%
runny nose	98%	98%	98%
no symptoms	97%	92%	94%

[1]Our data sets can be shared with interested parties upon request to avoid copyright issue. Please send us an email if you want to use our data sets.

[2]Our implementation in the actual Android devices is on-going.

B. *Differentiating Dry and Wet Cough*

We manually labeled 492 cough samples, out of which 323 samples (214 dry and 109 wet cough samples) were randomly picked for model training, and the remaining 169 samples (105 dry and 64 wet cough samples) were used for testing. For each sample, filtering was done to remove unwanted noise using some band pass filters and features were extracted from these samples. The features included 20 MFC coefficients, formant frequencies (F1, F2, F3), bi-spectrum score, log energy, kurtosis, ZCR and pitch. Parameter tuning for LR, RF and SVM models was performed using grid search with 10-fold cross validation.

After performing parameter tuning, the experimental results based on the three models LR, RF, and SVM are shown in Table III. We can observe that, the classification using the LR model can achieve the highest accuracy which is 93-94%. The regularization parameter used here is C=1 and the penalty is '$l2$' (Ridge Regression). Table IV shows the results of classifying dry/wet cough events using the LR model. We can observe that: 1) For dry cough, all the TPR, PPV and F1 score are more than 90%. 2) For wet cough, both the TPR and F1 score are more than 90%, but the PPV is smaller than 90% because the number of samples for wet cough are less and the corresponding false positives are larger.

TABLE III
DRY/WET COUGH CLASSIFICATION USING DIFFERENT MODELS

Model Type	Performance metrics			
	Accuracy	*TPR*	*PPV*	*F1 score*
LR	93-94%	94%	95%	94%
RF	86-88%	86%	86%	86%
SVM	83-85%	84%	84%	84%

TABLE IV
CLASSIFYING DRY/WET COUGH EVENTS USING THE LR MODEL

Type of Event	Performance metrics		
	TPR	*PPV*	*F1 score*
Dry cough	92%	99%	96%
Wet cough	98%	87%	92%

We also measured the time taken for training and testing the data, which is quite small, i.e., around 0.007s for training and 0.014s for testing.

V. DISCUSSION

Accuracy of locations. Accuracy of GPS locations usually depends on where the person is located, e.g., if the person is out and can see the open sky, the accuracy could be around 16ft. In addition, recent advancements in both the hardware and the standards make one-meter accuracy possible in Android devices. This makes it feasible for having an accurate location needed in our approach.

Broader applications of our approach. The symptoms like cough/sneeze/runny nose are pretty common for airborne diseases. For example, SARS (Severe Acute Respiratory Syndrome) is a virus transmitted through droplets that enter the air when a patient coughs or sneezes; one of symptoms of MERS (Middle East Respiratory Syndrome) is cough. Therefore, our approach is not just applicable to COVID-19, but is also applicable to any types of airborne epidemics or pandemics in the future.

Limitations and future work. The proposed approach in this paper has some limitations which will be further investigated in our future work. First, the current approach needs to rely on an assumption that all the participants are honest, e.g., they will not fake a coughing/sneezing/runny nose event to disturb the network. This assumption may not be true in practice and needs to be relaxed by introducing a flaw detection [12]. Second, the current design is purely centralized, and is good for a small-scale application (e.g., a local community, small town/city), in which a few central servers are enough to handle all the participants. However, a potential direction of scaling the applications will be transforming it to a fully distributed manner in which no central servers are required. Third, each smart device should be also capable of assessing the risk of virus exposure based on the events received as well as providing guideline to the user; in addition, to avoid overwhelming a mobile device, each event should only be broadcasted to those participants who are currently located near the event location. Fourth, we currently only consider cough/sneeze/runny nose for detection. Additional symptoms like fever may also help to improve the detection accuracy. There is a non-invasive method for detecting the body temperature from images captured by a thermal camera; optionally, we can measure the body temperature using temperature sensors if equipped in a mobile device.

VI. RELATED WORK

Table V lists previous studies in this topic and provides a comparison between them and our proposed work. The approaches proposed in [6], [10], [11], [13] simply focus on cough detection and do not incorporate networking/GPS locations to further prevent diseases spreading. The approaches in [5], [15] only focus on differentiating wet and dry cough, but the detection accuracy is very low and, most importantly, they do not incorporate networking and hence cannot prevent COVID-19 spreading. The work proposed in [14] includes detection of cough/sneeze/sniffle/throat clearing with the help of smartphone app using SVM-based multi-classification. Feature extraction is performed from the audio signals captured by the microphone of the smartphone and some of the no symptom signals are discerned using a threshold at the starting stage. However, what they have proposed still cannot prevent spreading of COVID-19 since event locations are not distributed to the network; in addition, their detection accuracy is low (i.e., 83%-88%); further more, the inference model has not been mentioned in their research. In our proposed work, we have mainly focused on detection of COVID-19 (or similar diseases) symptoms, including coughing/sneezing/runny nose, with the help of smartphone mic; we have evaluated several

machine learning models and selected the best among them as the final model; we have also considered how to differentiate dry and wet cough; in addition, we have used on-device inference, using TensorFlow Lite for deploying our models in Android devices and detecting events locally; finally and most importantly, the detected symptoms events along with the corresponding GPS locations (captured from the mobile devices) will be distributed to the network to alert potential victim people, preventing COVID-19 spreading.

TABLE V
A COMPARISON BETWEEN EXISTING WORK AND OUR WORK

Ref	Device used	Events detected	Model	Accuracy	Mobile Inference	Sending Alerts
[6]	Smartphone Mic	cough	CNN	85-90%	on-device inference	No
[11]	Smartphone Mic	cough	SVM kNN	80-81% 94-95%	Using Separate Systems	No
[13]	Microphone	cough	Ensemble classifier	90-92%	No	No
[10]	Cough detector	cough, cough-related-diseases	CNN	93-94%	cloud-based inference	No
[5]	No	dry/wet cough	Fuzzy c-Mean clustering	76-77%	No	No
[15]	No	dry/wet cough	LR model	80-85%	No	No
[14]	Smartphone Mic	cough sneeze sniffle throat-clearing	SVM	83-88%	not mentioned	No
Our work	Smartphone Mic	cough sneeze, runny nose, dry/wet cough	RF and LR classifiers	92-94% and 93-94%	on-device inference using TensorFlowLite	Yes

VII. CONCLUSION

In this work, we propose a novel approach to mitigate COVID-19 spreading via mobile devices broadly used by people today. Especially, the mobile devices can automatically detect symptoms events caused by the COVID-19 or similar diseases nearby, and distribute those events to people in the network, preventing them from getting exposed to potential viruses. The symptoms we consider include coughing, sneezing, and runny nose which may produce virus in the air. We have conducted experimental evaluation based on real-world data sets, which justifies that our approach can detect aforementioned symptoms with a high accuracy. We also discuss a few technical details on implementing the proposed approach into the Android devices.

REFERENCES

[1] Covid-19 myth busters. Retrieved March 30, 2020, from https://health.mcleancountyil.gov/723/COVID-19-Myth-Busters.

[2] Iceland lab's testing suggests 50% of coronavirus cases have no symptoms. Retrieved March 30, 2020, from https://www.cnn.com/2020/04/01/europe/iceland-testing-coronavirus-intl/index.html.

[3] librosa in github. Retrieved August 15, 2020, from https://github.com/librosa/librosa.

[4] Tensorflow lite inference. Retrieved August 14, 2020, from https://www.tensorflow.org/lite/guide/inference.

[5] Y. A. Amrulloh, D. A. R. Wati, F. Pratiwi, and R. Triasih. A novel method for wet/dry cough classification in pediatric population. In *2016 IEEE Region 10 Symposium (TENSYMP)*, pages 125–129, May 2016.

[6] F. Barata, K. Kipfer, M. Weber, P. Tinschert, E. Fleisch, and T. Kowatsch. Towards device-agnostic mobile cough detection with convolutional neural networks. In *2019 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 1–11, June 2019.

[7] MSS Safya Bhore and MS Shah. A comparative study of formant estimation. *International Journal of Advance Research in Electronics and Communication Engineering*, 4(12):2879–2882, 2015.

[8] Xiangfeng Dai, Irena Spasic, B. Meyer, Samuel Chapman, and Frederic Andres. Machine learning on mobile: An on-device inference app for skin cancer detection. 06 2019.

[9] T. Guo. Cloud-based or on-device: An empirical study of mobile deep inference. In *2018 IEEE International Conference on Cloud Engineering (IC2E)*, pages 184–190, April 2018.

[10] Ali Imran, Iryna Posokhova, Haneya N. Qureshi, Usama Masood, Sajid Riaz, Kamran Ali, Charles N. John, Muhammad Nabeel, and Iftikhar Hussain. AI4COVID-19: AI Enabled Preliminary Diagnosis for COVID-19 from Cough Samples via an App. *arXiv e-prints*, page arXiv:2004.01275, April 2020.

[11] Jesus Monge-Alvarez, Carlos Hoyos-Barcelo, Paul Lesso, and Pablo Casaseca-de-la Higuera. Robust detection of audio-cough events using local hu moments. *IEEE journal of biomedical and health informatics*, 23(1):184—196, January 2019.

[12] S. Rezaei, H. Radmanesh, P. Alavizadeh, H. Nikoofar, and F. Lahouti. Automatic fault detection and diagnosis in cellular networks using operations support systems data. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pages 468–473, 2016.

[13] Gowrisree Rudraraju, ShubhaDeepti Palreddy, Baswaraj Mamidgi, Narayana Rao Sripada, Y. Padma Sai, Naveen Kumar Vodnala, and Sai Praveen Haranath. Cough sound analysis and objective correlation with spirometry and clinical diagnosis. *Informatics in Medicine Unlocked*, 19:100319, 2020.

[14] Xiao Sun, Zongqing Lu, Wenjie Hu, and Guohong Cao. Symdetector: Detecting sound-related respiratory symptoms using smartphones. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, page 97–108, New York, NY, USA, 2015. Association for Computing Machinery.

[15] V. Swarnkar, U. R. Abeyratne, Y. A. Amrulloh, and A. Chang. Automated algorithm for wet/dry cough sounds classification. In *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3147–3150, 2012.