Text and Language independent classification of voice calling platforms using deep learning

Tapas Chakraborty*, Rudrajit Bhattacharyya, Priti Shaw, Sourav Kumar, Md Mobbasher Ansari, Mita Nasipuri and Subhadip Basu

Abstract Audio and video conferencing apps like Google meet, Zoom, Mobile call conference are becoming more and more popular. Conferencing apps are used not only by professionals for remote work, but also for keeping social relations. Present situation demands understanding of these platforms in details and extract useful features to recognise them. Our research focuses on collecting audio data using various conferencing apps. Audio data are collected in real world situation, i.e. in noisy environments, where speakers spoke in conversational style using multiple languages. After data collection, we have examined whether platform specific properties are present in the audio files or not. Pre-trained CNN models (Densenet, Resnet) are used to extract features automatically from the audio files. High recognition accuracy (99%) clearly indicates that these audio files contain significant amount of platform specific information.

Key words: Voice calling platforms, Audio conferencing, Google meet, Zoom, Discord, CNN, DenseNet, ResNet, Signal Processing

1 Introduction

During the pandemic, use of audio and video conferencing apps increases significantly. Conferencing apps are used not only for remote work or distance education, but also for social relations.

Zoom and Google meet are more popular than the other conferencing apps. In this research, we have also considered mobile call conferencing app and Discord. Zoom is a video conferencing software developed by ”Zoom Video Communica-

Tapas Chakraborty*, Rudrajit Bhattacharyya, Priti Shaw, Sourav Kumar, Md Mobbasher Ansari, Mita Nasipuri, Subhadip Basu
Jadavpur University, Kolkata, e-mail: ju.tapas@gmail.com, rudrajitb24@gmail.com, pritishaw0103@gmail.com, svkm240471@gmail.com, mobby0022@gmail.com, mita.nasipuri@jadavpuruniversity.in, subhadip.basu@ jadavpuruniversity.in

tions". It offers several packages to its customers. Free plan allows conference call up to hundred participants at a time, and maximum time limit of forty minutes. Whereas paid plan supports up to thousand participants lasting up to thirty hours. Google Meet (also called Hangouts Meet) is another video conferencing app developed by Google. Some of the features that both Zoom and google meet offer, are given below: (a) Multi-way audio and video calls (b) chat between participants (c) join through a web browser or through mobile apps (d) Screen-sharing (e) Join using dial-in numbers.

Discord is another calling and messaging platform popular among a gaming group, or friends who want to spend time together. Here users can communicate with each other using voice or video calls, text messaging, send media and files.

Audio data analysis from various conference calling platform requires a considerable amount of data. However, such data is not readily available. So we have decided to collect the data first then perform the analysis. In this experiment, audio data were collected using various conferencing apps. Speakers spoke in conversational style using multiple languages and in real world situation, i.e. in noisy environments. Audio data is recorded using one of the devises participating in the conference call.

Our focus is to extract platform specific features from the audio files. CNN has the capability to extract useful features automatically. Hence, CNN model is used for this purpose. Audio data is first pre-processed to remove noise and non-voice parts as described in [2]. Then Spectrograms are generated from those processed audio signals. We have used **Librosa**, a python library, [6] for audio data pre-processing and generating Spectrograms. Spectrograms are given as input to CNN model. During training phase, CNN is trained using audio files of known platforms. During testing, model is tested with audio files from unknown platforms.

Below is the block diagram of overall process.

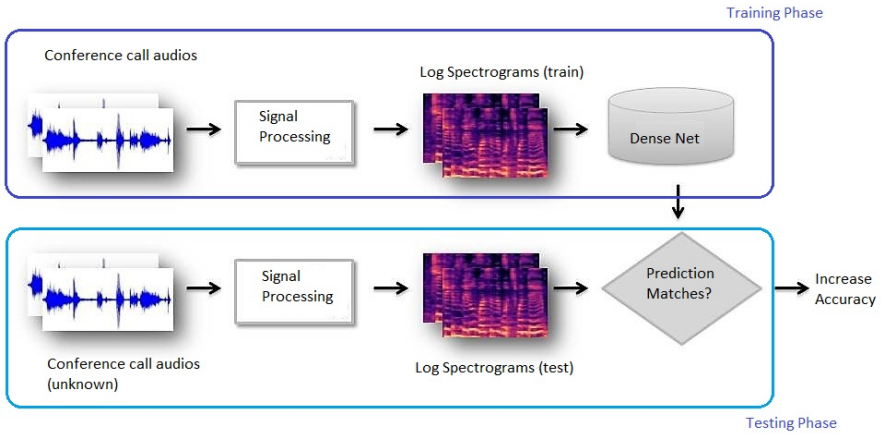

Fig. 1 A block diagram of overall process

Rest of the paper is organized as follows:- Data collection process is described in section two. Third section talks about various audio pre processing methodologies, CNN architecture used for classification. Experiments, results and conclusions are discussed in the remaining sections.

2 Data Collection

Audio data was collected mainly from the following four apps (a) Mobile conference call (b) Google Meet (c) Zoom (d) Discord

Two or more participants are talking to each other (a) in languages of their own choice (b) in conversational style where topics are not pre-defined (c) in real world i.e. environmental noises are present (d) speakers can speak anytime, that is why some overlapping conversations are there. Link of sample audio files given below provide link

There are 100 such audio files each of them having duration of approximately 5 minutes. Initially conference calls were recorded using device of one of the participants. Audio data have been used directly while Video data are converted into audio files. These audio files were then segmented into several 5 seconds audio clips using Audacity software. During this conversion, non-voice regions were removed as much as possible.

Figure 2 shows voice calling platform wise distribution of audio data

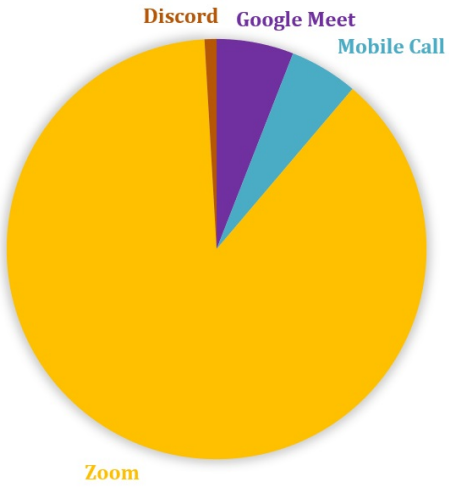

Fig. 2 Distribution of conference call recordings

3 Methodology

First step of this experiment is to determine the input to be provided for classification. As we know, processed data should not be used as input to a deep learning model like CNN, as model needs to extract features by itself. Therefore raw audio signal should be used as input. However, Dieleman et al. [3] showed that CNN performs better when Spectrograms are given as input to CNN model. So we have used Spectrograms as input to CNN. Audio signals are pre-processed first, to remove noise and silent parts. Then Spectrograms have been generated.

3.1 Pre-emphasis

Audio data have been passed through a high pass filter (HPF) to increase the amplitude of higher frequencies. This process removes low frequency noises as well. If $\mathscr{F}(t)$ is the audio signal, pre-emphasis can be done using below equation [2]

$$\mathscr{F}(t) = \mathscr{F}(t) - \alpha * \mathscr{F}(t-1) \tag{1}$$

Here α is a parameter. Usually value of α is chosen as 0.97.

3.2 Silence Frame Removal

Silence regions of the audio signal were removed by applying Short-Time Signal Analysis (STSA). Output of HPF i.e. pre-emphasized signal is divided into several time frames of short duration (20 ms window with overlap of 10 ms). Energy of those frames are compared with average frame energy of audio signal and silence frames were identified. [1]

$$K_{avg}(\mathbf{t}) = \Sigma_i(\frac{|f(\mathbf{t})^2|}{N}) \tag{2}$$

If $K_i > m * K_{avg}$ for a specific frame, then that frame is considered as voiced region. Otherwise it would be considered as silence frame and will be removed. Here, m is a parameter. Value of m was experimentally chosen as 0.2.

3.3 Spectrogram generation

Spectrograms has been generated from the pre-processed audio data. Figure 3 shows various Spectrograms generated from Zoom, Google meet, Mobile conference and

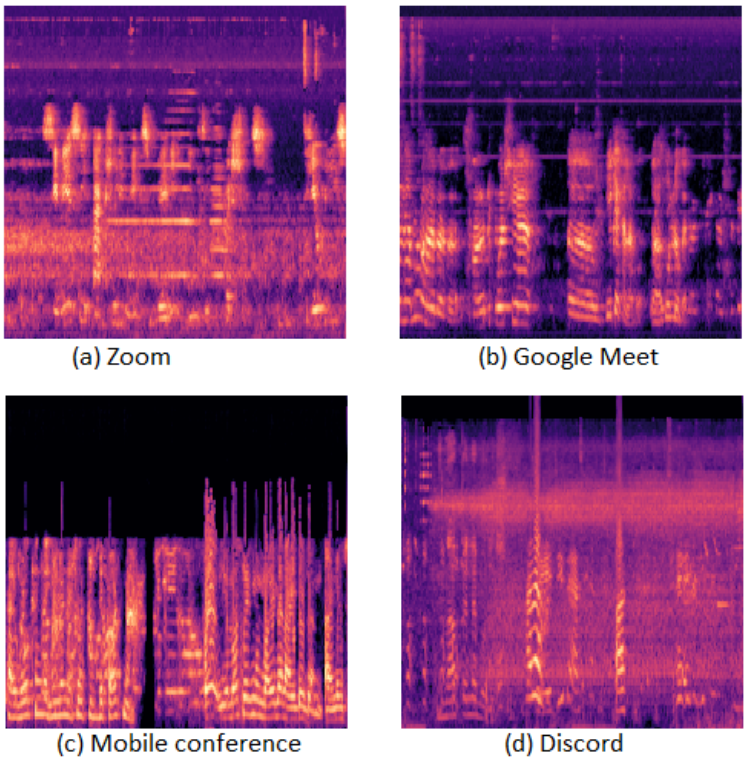

Fig. 3 Diagram of Mel-Spectrogram generated from Zoom, Google meet, Mobile conference and Discord respectively

Discord respectively. Below Spectrograms indicates that the audio information is significant enough to be fed into CNN network without further audio processing.

3.4 Model Architecture

Convolutional neural network (CNN) is a type of deep neural networks. CNN requires less pre-processing compared to other algorithms. This indicates that CNN has the capability to learn key features automatically.

Rather than building a CNN from scratch, models developed for other tasks can be used for our purpose. This technique is called Transfer learning, which is becoming more and more popular now a days. In this study, we have used two variations of such CNN models, DenseNet-201 [5] and Resnet-50 [4]. Figure-3 shows architecture of the model that we have used in this paper.

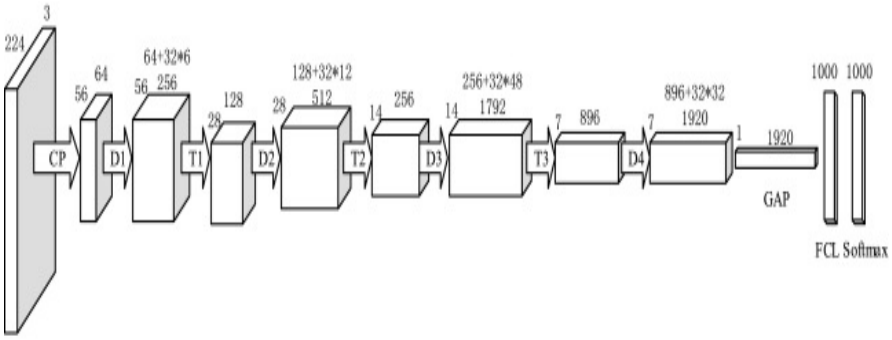

Fig. 4 Block diagram of DenseNet Model Architecture

4 Experiment and Results

4.1 Data preparation

Audio data collected from various voice conferencing apps are the source for this experiment. Hundred files are the source for this experiment, as described in Data collection section. These hundred files are sub-divided into 5 sec audio files and Spectrograms are generated from them. CNN usually requires balanced training data to ensure each class has equal contribution in the overall loss calculation. However, the data we collected, has unbalanced data. Some platforms have extremely short audio data, that makes this experiment more challenging.

Three fold cross validation method is applied on this data and recognition accuracy figures are noted.

4.2 Model Training

DenseNet-201 and ResNet-50 are implemented using tensor flow. During training, data was picked randomly to train the model. Training data was of the form (X_i, Y_i) where X_i is input data for i_{th} platform of shape 3x224x224 and Y_i is input label for i_{th} platform. Objective of the training is to minimize overall training loss with respect to all platforms.

Densenet201 model has been used with SGD optimizer and categorical cross entropy loss function. Model is trained on the training data set and validated on the validation data set. Accuracy is measured up to 150 epochs. Best model is determined by highest accuracy.

4.3 Identification using CNN

Let there are n platforms $\mathscr{S} = \{1, 2, 3, ..., n\}$. Output layer of DenseNet has n nodes, one for each platform. When an unknown audio data is given into the model, a vector with n scores will be returned as output. i^{th} scores indicate probability of that unknown audio to become to i^{th} platform. Maximum score is considered in this case. Decision rule of this identification process is given below

$$\hat{S} = \underset{k \in \mathscr{S}}{\arg\max}(p(\mathbf{x}_i)) \tag{3}$$

Here $\hat{S}$ is the identified platform and i^{th} platform's score is given by $p(\mathbf{x}_i)$, formula given below. The identified platform $\hat{S}$ has the maximum score.

$$p(\mathbf{x}_i) = \frac{e_i^x}{\sum_1^n e_k^x} \tag{4}$$

4.4 Performance Measure

The accuracy is measured by the percentage of correct identification, equation given below:

$$Accuracy(\%) = \Sigma_i(\frac{Numberofspeakerscorrectlyclassified}{Totalnumberof Speakers}) * 100 \tag{5}$$

Performance of this model was evaluated by confusion matrix. Confusion matrix is a table with two dimensions "actual" and "predicted", and similar sets of "classes" in both dimensions.

4.5 Results

Training and testing is done on audio data of three platforms. So we get three accuracy figures, one for each platform. We have reported overall accuracy as well.

Table 1 Overall accuracy and platform wise break up

Platform	DenseNet Accuracy	ResNet Accuracy
Zoom	99.76	99.92
Google Meet	100	100
Mobile conference	100	97.36
Discord	92.30	92.30
Overall	99.72	99.73

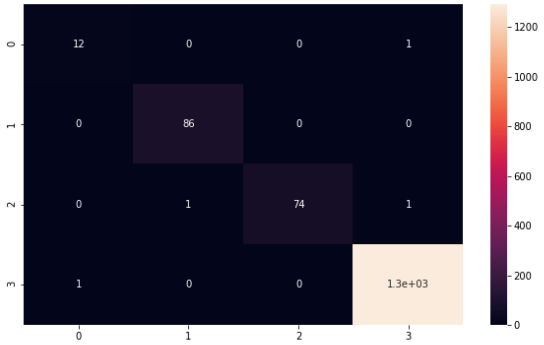

Fig. 5 ResNet Confusion Matrix

5 Conclusion

Main contribution of this research is to verify whether audio files from various conferencing platforms contains platform specific information or not. Data have been collected from various conferencing platforms frequently used today. Standard approaches have been followed to classify the audio files. High recognition accuracy indicates that audio files from various conferencing platforms retain platform specific information. Recognition of voice calling platform used for conference call will add value in forensic analysis.

In future, we are planning to collect audio data from other conferencing platforms like Skype, Whats-app, Facebook messenger, Microsoft Teams, Cisco Webex. Other methods will also be applied for improving classification accuracy.

Acknowledgements This project is partially supported by the CMATER laboratory of the Computer Science and Engineering Department, Jadavpur University, India, TEQIP-II, PURSE-II and UPE-II projects of Govt. of India. Subhadip Basu is partially supported by the Research Award (F.30-31/2016(SA-II)) from UGC, Government of India.

References

1. Barai, B., Das, D., Das, N., Basu, S., Nasipuri, M.: Closed-set text-independent automatic speaker recognition system using vq/gmm. In: Intelligent Engineering Informatics, pp. 337–346. Springer (2018)
2. Chakraborty, T., Barai, B., Chatterjee, B., Das, N., Basu, S., Nasipuri, M.: Closed-set device-independent speaker identification using cnn. In: Bhateja, V., Satapathy, S.C., Zhang, Y.D., Aradhya, V.N.M. (eds.) Intelligent Computing and Communication. pp. 291–299. Springer Singapore, Singapore (2020)

3. Dieleman, S., Schrauwen, B.: End-to-end learning for music audio. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 6964–6968. IEEE (2014)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
5. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2261–2269 (2017)
6. McFee, B., Raffel, C., Liang, D., Ellis, D.P., McVicar, M., Battenberg, E., Nieto, O.: librosa: Audio and music signal analysis in python (2015)