Speech Based Person Recognition for Biometric Applications

Garima Ranjan and M. Sabarimalai Manikandan
School of Electrical Sciences Indian Institute of Technology Palakkad
Email: 122101011@smail.iitpkd.ac.in and msm@iitpkd.ac.in

***Abstract*—This paper suggests a novel method for biometric application-related speech-based person recognition. Using a one-dimensional Convolutional Neural Network (1D CNN) trained on speech signals' Fourier Magnitude spectra, the uniqueness of speech based on frequency response is studied. The paper demonstrates 3 different models being trained and the results are compared. The dataset used in the paper is from Kaggle, Google Drive Dataset Folder**

I. Introduction

Speech-based person recognition is useful in biometric applications, i.e. user identification, without physical contact or special actions. Every person possesses a unique voice print which helps in enhancing security and accuracy, as speech is difficult to replicate or spoof compared to other biometric modalities. This makes the model a feasible option even for individuals with disabilities. The model can always be extended to work with other biometric models such as face recognition, for better security.

The techniques and technologies existing in the same field are as follows: In the text-based speaker detection system, the user is required to repeat a sentence, and based on that, they'll be classified. There are also text-independent systems that don't have any limitations on the sentences used for speaker recognition. Hybrid system, including both text-dependent and text-independent, are also implemented for better accuracy and reliability. Some models are purely based on voice characteristics such as pitch, tone, frequency, and speech patterns to create a unique voiceprint for each individual. For identifying speakers based on their voice characteristics by comparing them with stored voice samples, techniques such as Gaussian mixture models, neural networks, or dynamic time warping for matching are used. Models that identify specific keywords or phrases within speech for trigger actions or authentication are also there, these are often used in voice-controlled devices or virtual assistants such as Google Assistant. Statistical models representing speech patterns and acoustic characteristics are combined with language modeling techniques for context-aware speech recognition. Deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are increasingly used for speech recognition and speaker verification. These methods learn features directly from the speech signal, leading to improved accuracy. Features such as Mel-frequency cepstral coefficients (MFCCs), linear predictive coding (LPC), or spectrograms are extracted from speech to train most of the models for better results. Cloud-based APIs provided by companies like Google, Amazon, and Microsoft offer speech recognition and speaker identification capabilities, allowing developers to integrate speech recognition functionalities into their applications with ease.

Problems and Statements:
Speech properties can vary based on factors like accent, language, age, gender, and health conditions can affect the accuracy of recognition systems. Not all voice signals are clean audio, there are a variety of background noises, that can or cannot be removed from the given voice sample, causing difficulty in identifying the desired signal. Lack of diverse and annotated speech datasets for training recognition models, especially for underrepresented languages or dialects. Spoofing the legitimate speaker to get access, leads to privacy concerns. Perceived security, convenience, and usability are some of the factors that may affect user acceptance of speech-based authentication systems.

Motivation:
This project aims to create a robust and efficient speech-based person recognition system for biometric applications. By analyzing the signal's frequency domain properties, i.e., the Fourier magnitude spectrum, and training using 1D CNN, the objective is to enhance accuracy. This project addresses the need for authentication, involving signal processing techniques like Fast Fourier Transform(FFT) and deep learning models like 1D CNN to create a cutting-edge solution.

Major Objectives and Work Plan:
The major objective of this project is to study the signal properties based on the Fourier Magnitude Spectrum, which can be calculated using one-sided FFT. Based on these features, the 1D CNN model is trained for classification problems. The model is trained over several voice samples and mapped with the speaker. On testing with different voice samples for these speakers, the model is tested. The work plan follows the same objective, First, the speech signals will be processed to fetch the Fourier magnitude spectrum details. The speakers will be label-encoded for training purposes. The 1D CNN model will be trained for the same.

II. Materials and Methods

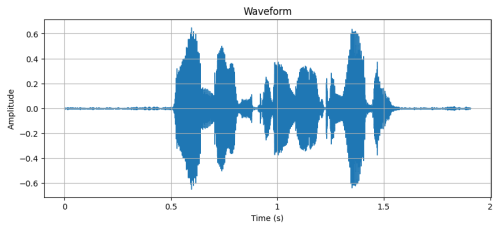

Fig. 1. Clean Signal example without windowing

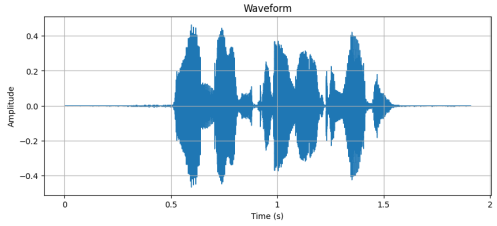

Fig. 2. Clean Signal example with windowing

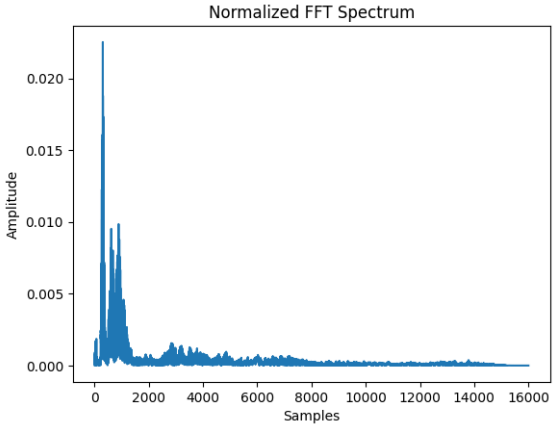

Fig. 3. FFT of example clean signal without windowing

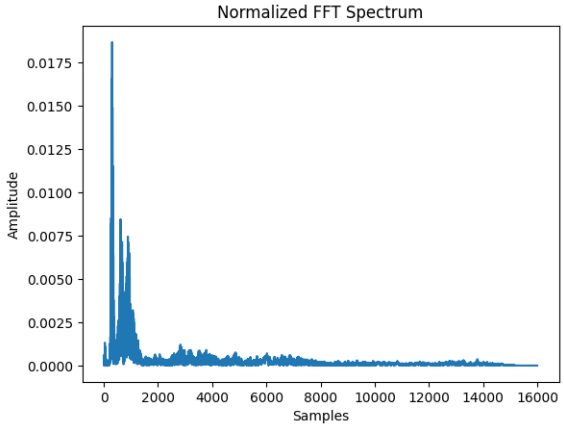

Fig. 4. FFT of example clean signal with windowing

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_8 (Conv1D)           (None, 23999, 64)         256

 max_pooling1d_8 (MaxPooling  (None, 11999, 64)        0
 1D)

 conv1d_9 (Conv1D)           (None, 11997, 128)        24704

 max_pooling1d_9 (MaxPooling  (None, 5998, 128)        0
 1D)

 flatten_4 (Flatten)         (None, 767744)            0

 dense_8 (Dense)             (None, 64)                49135680

 dense_9 (Dense)             (None, 18)                1170

=================================================================
Total params: 49,161,810
Trainable params: 49,161,810
Non-trainable params: 0
_________________________________________________________________
```

Fig. 5. 1D CNN model

A. System Architecture with Description

The dataset contains the speech file path and speaker name, the speech is loaded using librosa library. The data is augmented, using various techniques like stretching along time, shifting along time, adding noise, shifting pitch. The speech is resampled to a 16000Hz sampling rate and 3s duration. The Fourier magnitude spectrum for all speech signals is calculated using a one-sided Fast Fourier transform (FFT). The spectrum is interpolated to a fixed number of frequency samples (24001 in this case) to ensure consistency across signals and magnitude normalization is performed to scale the spectrum amplitudes. First, a machine learning model, logistic regression is applied to the dataset to train the model, and the accuracy is tested and compared to the deep learning model. For the training deep learning model, the training data is split into train, validation, and test data; the split ratio is set to 70 percent for training, 15 percent for validation, and 15 percent for testing. The Convolutional Neural Network (CNN) model is constructed using TensorFlow's Keras API. The model architecture consists of: Two 1D convolutional layers (Conv1D) with 64 and 128 filters, respectively, and ReLU activation. Two max-pooling layers (MaxPooling1D) to downsample the feature maps. A flattening layer is used to convert the 2D feature maps into a 1D vector. Two fully connected dense layers (Dense) with 64 and the number of output classes (for softmax activation) neurons, respectively. The model is compiled with the Adam optimizer and the categorical cross-entropy loss function. The model is trained using the fit method on the training data. Training is performed for 20 epochs with a batch size of 32.

Terminologies used:

Logistic Regression: Statistically, logistic regression is used

for binary classification tasks, where the output variable has two discrete values that usually reflect class labels like ”yes” or ”no,” ”spam” or ”not spam,” and so on. Logistic regression is not a regression algorithm, despite its name. It is a classification algorithm. It uses a logistic function, which is also called the sigmoid function, to express the likelihood that a case belongs to a certain class. Newton's method, maximum likelihood estimation, gradient descent, and other optimization techniques are used in logistic regression to estimate the parameters of the logistic function.

1D CNN: An 1D Convolutional Neural Network (CNN) is a type of deep learning architecture that is meant to look at sequential data that only has one dimension, like time series, signals, and text. Traditional CNNs used for image processing work with two-dimensional input data, like pictures. 1D CNNs, on the other hand, apply convolutional filters along the time axis of the input sequence. By swiping over the input sequence, these filters are able to identify local relationships and patterns. Translational invariance and hierarchical representations in sequential data are particularly well-captured by 1D CNNs. Applications for them include sentiment analysis, time series forecasting, audio categorization, and voice recognition.

Fourier Transform: To break down a function (or signal) into its individual bands, the Fourier Transform is used in mathematics. It represents the function as a sum of sinusoidal waves of varying frequency, each contributing to the total signal. Many disciplines, such as physics, engineering, and data analysis, employ the Fourier Transform extensively for processing and analysing signals and data in the frequency domain.

Fourier Magnitude Spectrum: The magnitude (or absolute value) of a signal's Fourier Transform is known as the Fourier Magnitude Spectrum. It depicts how signal energy is distributed among various frequencies. The magnitude spectrum highlights major frequencies and spectral features, offering important insights about a signal's frequency content. It is essential for tasks like filtering, feature extraction, and spectral analysis for analysts to be able to recognise significant frequency components and patterns within a signal by visualising the magnitude spectrum.

FFT: An effective method for determining the Discrete Fourier Transform (DFT) of a series of data points is the Fast Fourier Transform (FFT) algorithm. It uses symmetries and redundancies in the calculation to greatly speed up the Fourier Transform computation, especially for large datasets. With O(n log n) time complexity, the Fourier Transform of a sequence can be computed using the FFT technique, which makes it useful for high-throughput and real-time signal processing applications.

Windowing: In signal processing applications, such as speech recognition, the Hamming window is a frequently utilised windowing mechanism. Windowing is the process of applying a mathematical function to a signal, usually before the Fast Fourier Transform (FFT) or other procedures are carried out. Using this method lowers spectral leaks and other errors that might happen when looking at a short part of a signal. When the FFT is applied without windowing to a finite slice of a signal, spectral leakage happens. This may cause the spectral components to become smeared, which would make it more difficult to distinguish the different frequencies in the signal.

B. Mathematical Expressions and Equations

1) Equations for Fast Fourier Transform (single-sided):

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j \cdot \frac{2\pi}{N} nk} \tag{1}$$

$$MagX(k) = |X_k| \tag{2}$$

$$X_{norm}(k) = \frac{X(k)}{|X(k)|} \tag{3}$$

2) Hamming Window:

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \tag{4}$$

3) ReLu activation function:

$$\sigma(x) = max(0, x) \tag{5}$$

4) 1D CNN:

$$y[n] = \sum_{i=1}^{N} \sigma(x * w_i + b_i) \tag{6}$$

C. User Interface

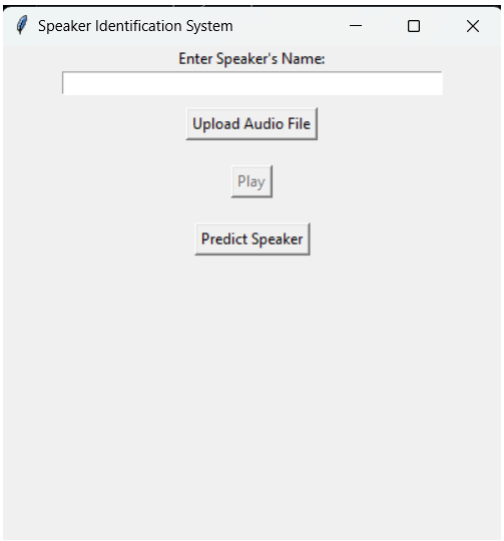

Fig. 6. Graphical User Interface

Section for Uploading Audio Files: This section enables users to upload audio files by using the ”Upload” button. The user can choose an audio file (in MP3 or WAV format) when the button is clicked. A file dialogue box then appears. The entry field next to the ”Upload” button displays the path of the selected file. Enter Speaker's Name Section: This section offers a box for the user to enter the speaker's name for verification. To make the prediction, the user must first enter the speaker's name. When a user uploads an audio file

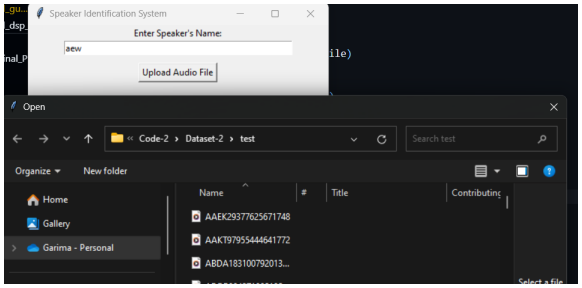

Fig. 7. Uploading Audio File

and enters the speaker's name, the Predict Speaker Button starts the speaker verification procedure. Upon clicking, the audio file is loaded, preprocessed, features (Fourier magnitude spectrum) are extracted, and the pre-trained model is used to make predictions. The prediction's outcome, which indicates if the projected speaker and the input name match, is shown beneath the button. Display of Results: The prediction result is

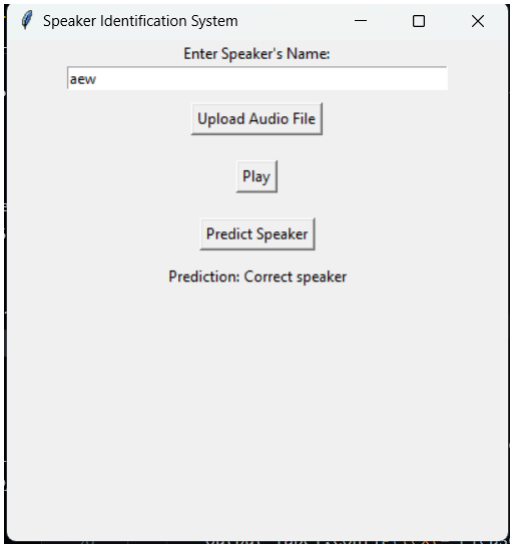

Fig. 8. Results shown on Graphical User Interface

shown on a label located beneath the "Predict Speaker" button. Prediction: "Correct speaker" appears on the label if the anticipated speaker and the input name match. Prediction: "Incorrect speaker" appears on the label if the expected speaker does not match the input name. Error messages are provided to notify the user in the event that any issues occur during the prediction process (such as an incorrect audio file or a processing failure). Error Handling: To provide a seamless user experience, the interface has error handling. Error warnings encourage the user to take proper action if they attempt to predict without uploading an audio recording or entering the speaker's name. Error notifications are also shown to the user in the event that any unforeseen errors (such as file loading failures or feature extraction difficulties) arise throughout the prediction process. Usability: The user is guided step-by-step through the speaker verification process via an intuitive and user-friendly interface. It is simple for the user to comprehend the necessary steps and submit the relevant data thanks to clear labeling and input fields. Enhancing usability and transparency, feedback messages and result display make sure the user gets timely input on the prediction outcome.

D. Results and Discussions

1) Dataset Collection: The following dataset is taken from kaggle, the dataset consist of 12000+ samples for training and testing. The dataset contains two CSV files with the file path to the audios and name of speaker, also contains .wav audio files.

```
          id                  file_path
0  rxr_a0591  train/rxr/arctic_a0591.wav  \
1  rxr_a0403  train/rxr/arctic_a0403.wav
2  ljm_a0059  train/ljm/arctic_a0059.wav
3  jmk_a0134  train/jmk/arctic_a0134.wav
4  rms_b0067  train/rms/arctic_b0067.wav

                                          speech speaker
0                We are both children together.     rxr
1  His newborn cunning gave him poise and control.     rxr
2             His immaculate appearance was gone.     ljm
3             He obeyed the pressure of her hand.     jmk
4  Below him the shadow was broken into a pool of...     rms
```

Fig. 9. CSV file for data

2) Experimental Results: From the available 12000 dataset values,the model is trained only on 3000 data values, this is due to memory constraints over the local system. The following training data was trained using 3 approaches. Basic Machine learning based algorithm, logistic regression, this gave the best results for these many datapoints. The second approach was to feed FFT based magnitude spectrum as feature vector to 1D CNN model, this also performed fairly well. The third approach was to use windowing before calculating FFT, to reduce spectral leakage, the overall performance compared to other two approaches was worse in this case.

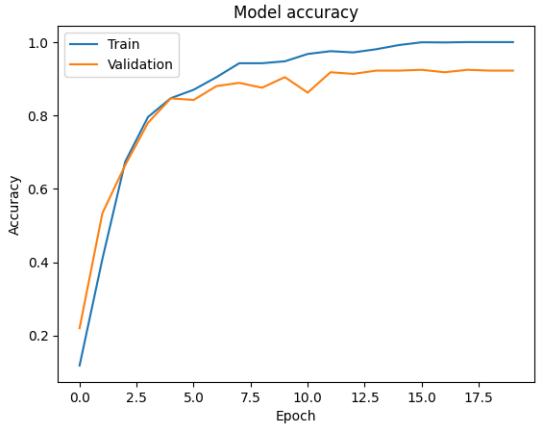

Fig. 10. 1D CNN model accuracy for non-windowing

E. Classification report

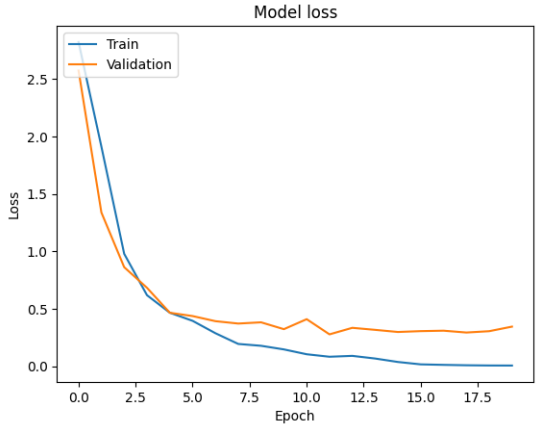

Fig. 11. 1D CNN model loss for non-windowing

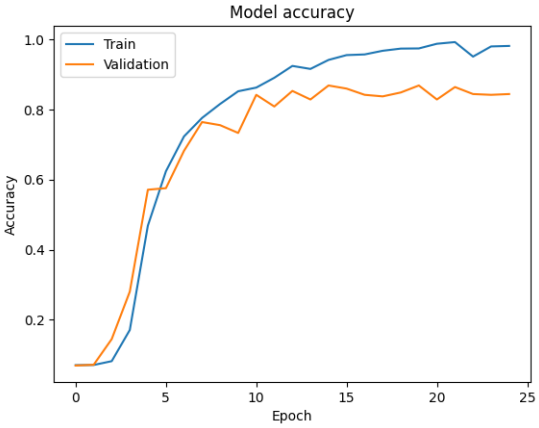

Fig. 12. 1D CNN model accuracy for hamming windowing

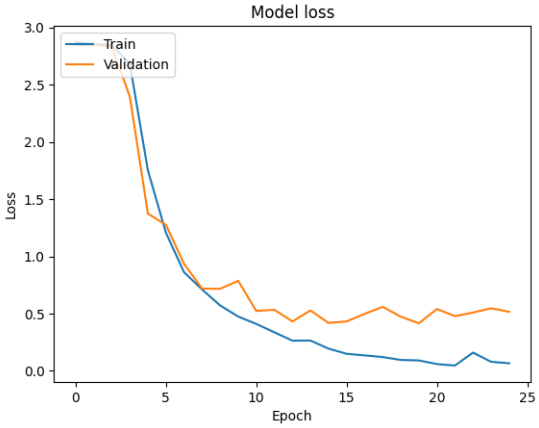

Fig. 13. 1D CNN model loss for hamming windowing

Training Type	Window	Number of data points	Number of epochs	Train %	Train Loss	Validation %	Val loss	Test %
1D CNN	None	3000	20	100	0.0058	92.22	0.3447	90.1
1D CNN	Hamming	3000	25	98.19	0.0667	84.44	0.5163	88.33
Logistic regression	None	3000	-	-	-	-	-	91.5

Fig. 14. Comparing the three models proposed

Training Type	Window	Accuracy	Sensitivity (macro avg)	Specificity (macro avg)
1D CNN	None	0.90	0.88	0.88
1D CNN	Hamming	0.88	0.87	0.87
Logistic regression	None	0.92	0.90	0.90

Fig. 15. Comparing the accuracy, specificity, and sensitivity of three models proposed

```
[[172   4   2   1   7   3   1   6   5   0   1   2   0   4   4   4   2   1]
 [  0 106   0   0   0   0   0   0   1   0   0   4   0   0   0   2   0   0]
 [  0   0 112   0   4   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0 213   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0 111   0   0   1   0   0   0   0   0   1   0   0   1   1]
 [  1   1   3   7   1 195   0   2   1   0   2   1   0   3   0   0   0   0]
 [  0   0   0   0   0   1 211   0   0   0   0   0   0   0   0   0   0   8]
 [  0   1   0   0   1   3   1  90   0   0   0   0   0   3   2   0   0  12   1]
 [  0   0   0   0   1   0   0   0 113   0   0   0   0   0   0   0   0   0]
 [ 16   0   0   0   0   0   0   0   0  91   2   5   0   0   0   1   0   0]
 [  0   0   0   0   0   0   0   0   0   0 218   0   0   0   0   0   0   0]
 [  1   0   0   0   0   0   0   0   0   0   0 217   0   0   0   0   1   0]
 [  0   0   0   0   3   0   2   3   0   0   0   0   0  78   8   0   0  18   3]
 [  1   0   0   0   0   8   0   0   0   0   0   0   0   3 201   1   0   2   2]
 [  4   0   1   0   0   0   0   1   1   0   0   0   0   0   0 205   0   0   0]
 [  1   6   3   0   1   0   0   0   1   1   0   6   0   4   0   0 104   1   1]
 [  0   0   0   0   3   0   2   3   0   0   0   0   0   6   0   0   0 100   0]
 [  0   2   0   1   0   0   2   0   1   0   3   0   1   0   0   0   0   0 208]]
Accuracy 0.915
```

Fig. 16. Machine Learning: Logistic Regression Confusion Matrix

	class	specificity	sensitivity
0	0	0.991370	0.785388
1	1	0.995151	0.938053
2	2	0.996878	0.957265
3	3	0.996771	1.000000
4	4	0.992721	0.965217
5	5	0.994251	0.898618
6	6	0.997122	0.959091
7	7	0.994456	0.789474
8	8	0.996535	0.991228
9	9	0.999653	0.791304
10	10	0.997124	1.000000
11	11	0.993528	0.990868
12	12	0.995494	0.678261
13	13	0.992092	0.922018
14	14	0.998207	0.966981
15	15	0.997562	0.806202
16	16	0.987179	0.877193
17	17	0.993889	0.954128

Fig. 17. Machine Learning: Logistic Regression specificity and sensitivity for each speaker

F. Conclusions

From the results, we can see that logistic regression performs best out of the two CNN models, this might be due to less number of data points and clean audio signals. The windowed CNN performs comparatively lowest, this might be due to suppress of important information present in the audio

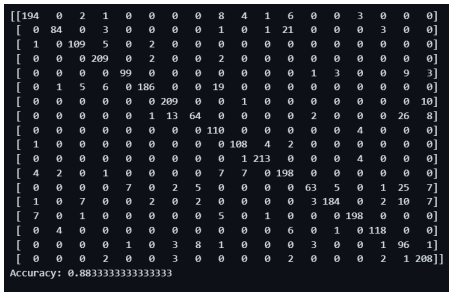

Fig. 18. 1D CNN Learning with windowing: Confusion matrix

	class	specificity	sensitivity
0	0	0.994966	0.885845
1	1	0.997575	0.743363
2	2	0.994797	0.931624
3	3	0.993541	0.981221
4	4	0.997227	0.860870
5	5	0.997485	0.857143
6	6	0.992446	0.950000
7	7	0.994802	0.561404
8	8	0.985100	0.964912
9	9	0.995494	0.939130
10	10	0.997484	0.977064
11	11	0.986695	0.904110
12	12	0.996880	0.547826
13	13	0.996765	0.844037
14	14	0.996055	0.933962
15	15	0.996865	0.914729
16	16	0.975398	0.842105
17	17	0.987060	0.954128

Fig. 19. 1D CNN Learning with windowing: specificity and sensitivity for each speaker

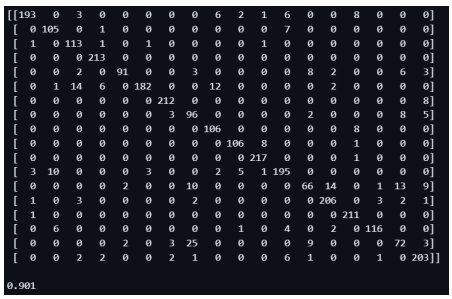

Fig. 20. 1D CNN Learning without windowing: Confusion matrix

signal while training due to windowing, as we are training over fourier magnitude spectrum. Although the three models have different accuracies, all three of them performs well as they give accuracy of more than 90 percent. The signal with windowing carries more relevant data.

G. Future Plans

The possible scope for future would be to take in account the noises that are unavoidable in real life scenarios, considering these noises, extracting the required waveform using designed

	class	specificity	sensitivity
0	0	0.997843	0.881279
1	1	0.994112	0.929204
2	2	0.991675	0.965812
3	3	0.996412	1.000000
4	4	0.998614	0.791304
5	5	0.998563	0.838710
6	6	0.997122	0.963636
7	7	0.985793	0.842105
8	8	0.993070	0.929825
9	9	0.997227	0.921739
10	10	0.996046	0.995413
11	11	0.991730	0.890411
12	12	0.993068	0.573913
13	13	0.992811	0.944954
14	14	0.993544	0.995283
15	15	0.998258	0.899225
16	16	0.989951	0.631579
17	17	0.989576	0.931193

Fig. 21. 1D CNN Learning without windowing: specificity and sensitivity for each speaker

filters for better accuracy and reliability. For real life implementation, the model can be trained on real time dataset from speakers rather than already available dataset.

REFERENCES

[1] Speaker Recognition - CMU ARCTIC (Kaggle, Dataset owned by GABRIEL LINS).
[2] Person Identification Using Face and Speech Recognition for Visually Challenged with Mask Detection.
[3] J. Gomes, H. Fernandes, S. Abraham and S. Chavan, "Person identification based on voice recognition," 2021 4th Biennial International Conference on Nascent Technologies in Engineering (ICNTE), Navi-Mumbai, India, 2021, pp. 1-5.