

Speaker Recognition and Detection of Liveliness of Voice in ATMs & Mobile Phones

A Dissertation Submitted to the University of Hyderabad in

Partial Fulfillment of the Degree of

Master of Technology

in

Computer Science

by

RITIK RAJ

18MCMT18



School of Computer and Information Sciences

University of Hyderabad

Gachibowli, Hyderabad-500046

June, 2020



CERTIFICATE

This is to certify that the dissertation titled, **Speaker Recognition and Detection of Liveliness of Voice in ATMs & Mobile Phones**, submitted by **RITIK RAJ**, bearing Regd. No. 18MCMT18, in partial fulfillment of the requirements for the award of Master of Technology in Computer Science is a bonafide work carried out by him under my supervision and guidance at the Centre for Mobile Banking(CMB), IDRBT during June 2019 to June 2020.

The dissertation has not been submitted previously in part or in full to this or any other University or Institution for the award of any degree or diploma.

Dr. V.N. Sastry
Project Supervisor,
Professor and Head(CMB),
IDRBT, Hyderabad

Dr. Chakravarthy Bhagvati
Dean
School of CIS,
University of Hyderabad

DECLARATION

I, **RITIK RAJ**, hereby declare that this dissertation titled, "**Speaker Recognition and Detection of Liveliness of Voice in ATMs & Mobile Phones** " submitted by me under the guidance and supervision of **Dr. V.N. Sastry** is a bonafide work which is also free from plagiarism. I also declare that this dissertation has not been submitted previously in part or in full to this University or other University or Institution for the award of any degree or diploma.

A report on plagiarism statistics from the University Librarian is enclosed.

Date: June 29th, 2020

//Countersigned//

Signature of the Student

Place: Hyderabad

(RITIK RAJ)

18MCMT18

Signature of the Supervisor

(Dr. V.N. Sastry)

ACKNOWLEDGEMENT

I would like to take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of my course of research.

I would like to express my sincere gratitude to **Dr. V. N. Sastry** Sir, my project supervisor, for valuable suggestions and keen interest through out the progress of my course of research.

I would like to thank **Dr. A. S. Ramasastry**, Director, **IDRBT** for providing me infrastructural facilities to work in, without which this work would not have been possible.

I am grateful to the Head, Center for Mobile Banking for providing excellent computing facilities and a congenial atmosphere for progressing with my project.

I would like to thank **Institute for Development and Research in Banking Technology, Hyderabad** for providing all the necessary resources for the successful completion of my course work.

At last, but not the least I thank my family members, classmates and other students of SCIS for their physical and moral support.

I would also like to thank the **Open Source Community** who provided the free Software and documentation to work with.

With Sincere Regards,
RITIK RAJ

ABSTRACT

Bio-metric authentication is rapidly increasing as a replacement of a traditional PIN-based authentication system. In bio-metric we analyze human unique characters like face, fingerprint, iris, a voice that is unique and does not show similarity with another human. Voice as a biometric authentication system because of its unique character and easy to use, but voice bio-metric has its disadvantages, it is very much vulnerable to replay attack; try to access using a recording of user's audio sample.

In this project work, We proposed a Text-Independent based Speaker-Recognition and it's liveliness detection based on Speaker's unique voice. In this project work, we used combine cepstral features likes MFCC & GFCC for differentiating speakers and used Doppler shift based features for detection of liveliness. We also reduced the audio data length for speaker recognition which will help us implement this authentication system in ATMs, Mobile Phone, and Banking Transaction.

Keywords – Speaker Recognition, Liveliness Detection, Text-Independent speech, Automatic Teller Machine, Convolution Neural Network

Contents

CERTIFICATE	i
DECLARATION	ii
ACKNOWLEDGEMENT	iii
ABSTRACT	iv
1 Introduction	1
1.1 Voice Biometric and User Authentication	1
1.1.1 Voice Biometric	1
1.1.2 User Authentication	1
1.2 Speaker Recognition	1
1.3 Liveliness Detection	2
1.4 Cepstral Features	3
1.5 Application	5
1.5.1 Speaker Recognition(SR)	5
1.5.2 Liveliness Detection	6
1.6 Project Objective	6
1.7 Problem Statement	6
1.8 Overview of Dissertation	7
2 Literature Survey	8
2.1 Bio-metric Authentication methods in ATM's and Smartphones	8
2.2 Speaker Recognition Technique	9
2.3 Liveliness Detection Technique	10
2.4 Convolution Neural Network(CNN)	11

3	Speaker Recognition using CNN	15
3.1	Data Description	15
3.2	System Flow	15
3.3	Speaker Recognition Algorithm	19
3.4	Illustration	20
3.5	Result	21
3.6	Conclusion	22
4	Speaker Liveliness Detection using Text-Independent Voice	23
4.1	Introduction	23
4.2	Data Collection	24
4.3	Flow Diagram	24
4.4	Liveliness Detection Algorithm	27
4.5	Result	29
4.6	Conclusion	30
5	Comparative Analysis of Various Text-Independent Voice Recognition Model	31
5.1	Introduction	31
5.2	Comparative Analysis	31
5.2.1	Performance of Models for different Audio Length	32
5.2.2	Performance Analysis based on Different ratio of overlapping frames	34
5.2.3	Performance observation under different Number of Speaker	35
5.3	Observation and Conclusion	36
6	Conclusion and Future Work	37
6.1	Conclusion	37
6.2	Future Work	37
A	Appendix	41
A.1	2 Speaker Confusion Matrix	41
A.2	3 Speaker Confusion Matrix	42
A.3	4 Speaker Confusion Matrix	42
A.4	5 Speaker Confusion Matrix	43

A.5	6 Speaker Confusion Matrix	43
A.6	7 Speaker Confusion Matrix	44
A.7	8 Speaker Confusion Matrix	44
A.8	9 Speaker Confusion Matrix	45

List of Figures

1.1	Calculating MFCC Block Diagram	3
1.2	Calculating GFCC Block Diagram	5
2.1	Convolution Neural Network	11
2.2	Filtres	12
2.3	Rectified Linear Unit	13
2.4	Max-pooling	13
3.1	Proposed Speaker Recognition System	16
3.2	Framing of Audio without overlapping	16
3.3	Process of conversion of 1D to 2D	17
3.4	CNN Architecture for Speaker Recognition	18
3.5	time domain audio data	19
3.6	1D converted audio data into 2D	20
3.7	Performance Plot Accuracy & Loss	21
3.8	Confusion matrix of Speaker Classification of 10 user	22
4.1	Block Diagram of Liveliness Detection	25
4.2	ANN architecture for Liveliness Detection	26
4.3	Input data for liveliness detection	27
4.4	Loss plot Liveliness Model	29
4.5	Confusion matrix of Liveliness model	30
A.1	Confusion Matrix of 2 Speaker	41
A.2	Confusion Matrix of 3 Speaker	42
A.3	Confusion Matrix of 4 Speaker	42
A.4	Confusion Matrix of 5 Speaker	43

A.5	Confusion Matrix of 6 Speaker	43
A.6	Confusion Matrix of 7 Speaker	44
A.7	Confusion Matrix of 8 Speaker	44
A.8	Confusion Matrix of 9 Speaker	45

Chapter 1

Introduction

In this chapter, we discuss on voice biometric, user authentication, speaker recognition, and their application, challenges, motivation, problem statement, and project scope.

1.1 Voice Biometric and User Authentication

1.1.1 Voice Biometric

It all started in 1867, with Alexander Melville Bell laid the research ground for the upcoming generation by inventing a language called Universal Alphabetics.

It can be defined as a system that leverages the sound of a human, which is produced by its articulatory to identify him/her. It uses a unique character of a user's voice to identify the user among a pool of speakers.

1.1.2 User Authentication

It can formally be defined as the process of identifying a user by the system so as to allow users to access services and applications. It is a necessary step to secure the user's work and data from unauthorized or evil intention users.

1.2 Speaker Recognition

It is defined as the potential of a machine and system to recognize the identity of the speaker by leveraging the unique nature of the speaker's voice. Thus, it is possible to use

it as a replacement of PIN-based or traditional methods of authentication and thereby can be used in Banking, Telecommunication, and Gadgets. The kind of voice features to be extracted would be discussed in later chapters. Before discussing categories it is better to have some clarity about enrollment procedure in the speaker recognition system. Basically, we provide our voice sample to the system, so that the system can extract unique features from samples.

Speaker recognition can be divided into two categories:-

- **Text-Dependent** It is defined as a word or set of words or sentence that is used to be read or spoken at the time of enrollment in the speaker recognition system, the same utterance of a word or set of words or sentence is needed to be spoken at the time of identification/verification. It is also memory-based user authentication based on bio-metric.
- **Text-Independent** It is defined as a word or set of words that is used at the time of enrollment in the speaker recognition system which need not be the same words to be spoken at the time of identification/verification.

1.3 Liveliness Detection

It is defined as the potential of a machine or system to differentiate between the live user and its impersonator. The impersonator may have the users' audio sample or image sample to bypass the system, which does not have a liveliness detection feature.

Approaches to Detect liveliness :

- **Human Body vibration:** When we speak, our body/throat vibrates with word utterance. Different words produce different vibrations of energy. Thus, by using vibration sensor, we can detect liveliness as in wearable device.
- **Audio-video Co-relation:** Every sound we produce also generates different facial expressions or facial movements, By detecting and analyzing such audio and facial expression movements one can detect liveliness.
- **Doppler Shift:** Doppler shift is a technique used in the radar system. As our articulatory system contains the lip, jaw, tongue when these moves multidimensionally, it produces a disturbance in the air. Thus analyzing these disturbances

can help detect liveliness. We discuss this type further in chapter 4. We used this method to detect liveliness on text-independent voice authentication.

1.4 Cepstral Features

Cepstral features [3] of voice try to replicate the process as a human perceives sound . Humans are sensitive to low-frequency sound as compared to high frequency. Around 80% of our audio receptors are used for hearing sound below 8KHz. In another sense, its logarithmic distribution of human receptors to hear the sound. Feature extraction is procedure to extract a set of features from the audio frames(audio signals are divided into a small unit known as a frame). We extract the cepstral feature from each frame. There are various cepstral features extraction method like, Gamma-tone frequency cepstral coefficient(GFCC), Linear predictive cepstral coefficient(LPCC), melody frequency cepstral coefficient(MFCC), Bark-frequency cepstral coefficient(BFCC). We focused on MFCC and GFCC as our feature extraction method for experiments.

Mel Frequency Cepstral Coefficient(MFCC)

Steps for calculating MFCC [3]

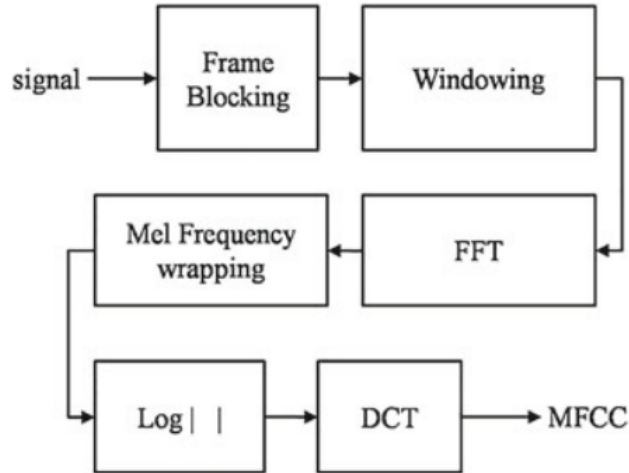


Figure 1.1: Calculating MFCC Block Diagram

Algorithm 1.1 MFCC(S(k))

1. Frame the time domain audio signal S(K) into Z small time domain audio signal $s_i(K)$.

where $s_i(K)$ is i^{th} frame, $0 \leq i \leq Z - 1$

2. Apply Discrete Fourier Transform(DFT) at each frame $s_i(K)$

$$S_i(K) = \sum_{n=1}^N s_i(K)h(n)e^{-j2\pi kn/N}; 1 \leq k \leq K \quad (1.1)$$

where $S_i(K)$ is DFT of i^{th} frame, $h(n)$ is Hamming window, N is window length, K is length of window

3. Calculate Power Spectrum at each $S_i(K)$

$$P_i(K) = \frac{\|S_i(K)\|^2}{N} \quad (1.2)$$

where $P_i(K)$ is power spectrum at i^{th} frame.

4. Take log of each power spectrum
 5. Apply Discrete Fourier Transform to result of step 4 to get MFCC co-efficient
-

Gamma-tone frequency cepstral coefficient(GFCC)

Steps for Calculating GFCC [8]

Algorithm 1.2 GFCC(S(t))

1. Speech signal S(t) multiplied by Gamma-tone filter bank $g(t)$ in frequency domain.

$$g(t) = a^{t-1}e^{-2\pi bt} \cos(2\pi f_c t + \phi) \quad (1.3)$$

$$ERB(f_c) = 24.7(4.37 \frac{f_c}{1000} + 1); b = 1.019ERB \quad (1.4)$$

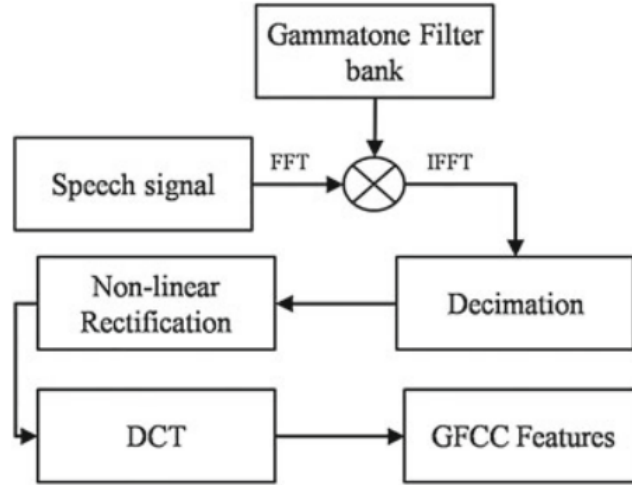


Figure 1.2: Calculating GFCC Block Diagram

$$s_r(t) = S(t)g(t) \quad (1.5)$$

where n is the order of filter, ϕ is the Phase Shift, f_c is the central frequency, b bandwidth.

2. Apply Inverse Fourier transform to result of $s_r(t)$
3. Decimate signal to 100Hz to remove noise
4. Apply DFT to resultant to get GFCC

1.5 Application

1.5.1 Speaker Recognition(SR)

SR in Biometric authentication has been gaining wide popularity over the past year due to the availability of computational power and with improving accuracy.

- **Security and Access Control :** Lots of smartphones and gadgets have speakers and microphones. Thus speaker recognition can be used as the fastest way to lock and unlock devices. In the future, it can be used in self-driving cars, lockers of houses, etc.

- **Payment** : In ATMs and Smartphones, speaker recognition, could be used as a replacement of traditional authentication systems with the added benefit of not remembering passwords anymore if the users voice itself is the authentication key.
- **Law Enforcement** : Speaker recognition can be used for identifying a culprit based on the available voice sample. Very handy to detect a person who uses call or recorded audio as a threatening tool.

1.5.2 Liveliness Detection

Liveliness Detection of the user from the presentation attack(Intentionally try to defect bio-metric security) is a necessity in today's data world. We giveaway our bio-metric data freely, and unknowingly on social networking sites. Anyone with wrong intention can misuse that data for malicious activity. So ensuring security of user's voice data security is also necessary.

- **Payment** : The presentational attacker may try to replay recorded audio samples for the transaction; with liveliness detection, we could avoid that.
- **Robustness to Bio-metrics** : Unlike most bio-metrics as face recognition, speaker recognition is vulnerable to presentation attack. With a newly added feature as liveliness detection, It would make these bio-metrics more robust to presentational attack.

1.6 Project Objective

To provide a comparative solution for Text-Independent Speaker Recognition with added features of detecting liveliness of the speaker.

1.7 Problem Statement

The three problems that had been studied are as follows:

- **Text-Independent Speaker Recognition** : Even with many research's are being carried out in Speaker recognition, the challenges faced haven't been solved entirely.

One of the challenges, which caught our attention was the problems caused due to dependency on the spoken word of the audio sample as well as length of audio for verification/identification.

- **Speaker Liveliness Detection using Voice :** The liveliness of the speaker detection is a typical problem associated with Speaker recognition, as we discussed in chapter 1.3. We used a Doppler's effect based feature to detect live users using a deep learning method.

The primary object of this project is to carry out text-independent speaker-recognition and detect the liveliness of the speaker's using only voice.

1.8 Overview of Dissertation

The rest of the dissertation is organized as follows:

- **Chapter 2:**
We discuss related previous work in the field of Speaker Recognition and Liveliness Detection as well as bio-metric inclusion in ATM's and Smartphones. We also had discussion about Convolution Neural Network.
- **Chapter 3:**
Explained about used Convolution Neural Network architecture, Data pre-preparation technique which solved the Speaker Recognition problem
- **Chapter 4:**
Explained about the technique of extraction Doppler shift data for Liveliness and proposed model to solve text-independent Liveliness Detection System.
- **Chapter 5:**
In this chapter, we try to compare our proposed model with other models with respect to audio length, the number of speakers, feature-length. Thus, we draw a conclusion based on our observation.
- **Chapter 6:**
Conclusion of the dissertation and future work of project.

Chapter 2

Literature Survey

In this chapter, we have discussed different methods of bio-metric authentication in ATM's and smartphones. Then the literature surveys on different approaches of text-independent Speaker Recognition and Liveliness Detection technique. Then, we discuss the Convolution Neural Network..

2.1 Bio-metric Authentication methods in ATM's and Smartphones

Bio-metric in ATM's and smartphones evolved over the years in terms of banking requirements. They are developed around banking-related authentication using bio-metric.

- **Finger Vein Reader Technology :**

Barclays, a financial services company, partners with technology giant Hitachi; which will develop Finger Vein reader technology to the customer for transaction authentication as a replacement of PIN or token system. The technology idea lies in the fact that the finger's vein does not change since the womb[2].

- **Voice Authentication :**

In 2016, CITI bank introduced voice biometric to authenticate customers for a call on the help center, which previously required account number or ATM card number on the mobile device. It saves customers a lot of time writing long sequences of numbers.

- **Fingerprint Technology :**

In 2019, The Royal Bank of Scotland(RBS) and NatWest announced a program that provides biometric authentication based cards, and in place of traditional authentication methods, customers can do a transaction using fingerprints at ATMs. In 2016, Bank of America offered its customers to do banking using the fingerprint sensor of mobile devices.

- **Iris Technology :**

Wells Fargo's Commercial Electronic Office[12] allowed its commercial customer to view balance, make a deposit and do all kinds of transactions using mobile devices using token generation apart from that as secondary authentication customers can use mobile's camera to scan eye and process transactions. In 2017, Bank of America, as a pilot program, allowed the customer to use their mobile device's camera to iris scan and do all kinds of transactions.

2.2 Speaker Recognition Technique

There are various research activities going on in the field of Speaker Recognition as we have discussed its type in earlier chapter 1.2. Our main focus was text-independent based feature Recognition.

Boles et al.[6] approaches text-independent Speaker Recognition using short-term feature extraction where it used MFCC as a cepstral feature on LibriSpeech dataset with ten Speakers got accuracy around 93.35% on the feed-forward network with three hidden layers and 128 neurons each.

Ahmad et al.[1]assesses for text-independent Speaker Recognition by using a probabilistic neural network on the VoxForge database with Speakers around five Speakers used DMFCC, and DDMFCC got around the accuracy of 94%.

2.3 Liveliness Detection Technique

Feng et al.[7] needs a wearable device that collects the body-surface vibration using the accelerometer of the Speaker and matches it with the audio sample received through a microphone. Due to its dependency on wearable gadgets make it less applicable to various places. It has good performance accuracy of around 97%.

Zhang, Linghan et al.[5] calculates time-difference arrival(TDoA) of the changes in the sequence of phonemes sound using two recording device of the smartphones and uses such a unique TDoA that doesn't exist during replay attacks for detection of live user. It achieves an accuracy of around 99% but with an ideal environment setup like it needed the same distance and position from smartphone and mouth as it was at the time of enrollment. Slight change in position or distance can decrease the accuracy or performance exponentially.

Wang, Qian et al.[10] uses pop noise, which is produced during breathing very close to the microphone. Later this pop noise was processed for Liveliness Detection. Voice pop can wrongly detect noise or background sound as pop noise. Accuracy decreases with an increase in the difference between device and articulator with the right environment achieved an accuracy of around 93%.

Zhang et al.[9] Linghan Zhang, Sheng Tan, et al.[5] detect a live user by using the unique articulatory posture of mouth and mobile's Speaker and microphone. It uses the device's loud-speaker as a Doppler shift, that continuously plays a high-frequency sound(around 15KHz) because of its unhearable to human ears. The Speaker listens to the reflected high frequency with the user's spoken passphrase. The audio signal reflection due to the user's unique articulatory gesture, which has multidimensional movement, causes the Doppler shift, which is analyzed for liveliness of live user detection. It is a test-dependent approach and robust to different hardware(Speaker, microphone) displacement, having detection accuracy around 99%.

2.4 Convolution Neural Network(CNN)

Convolution Neural Networks(CNN2D) are also based on Neural Networks with how humans visualize the image. It is primarily used for image classification; later, there are various versions of CNN which are used for Image segmentation, regression, classification. CNN 1D is now also used in speech Recognition, audio classification. In this dissertation work we interchangeably use CNN with Convolution Neural Network.

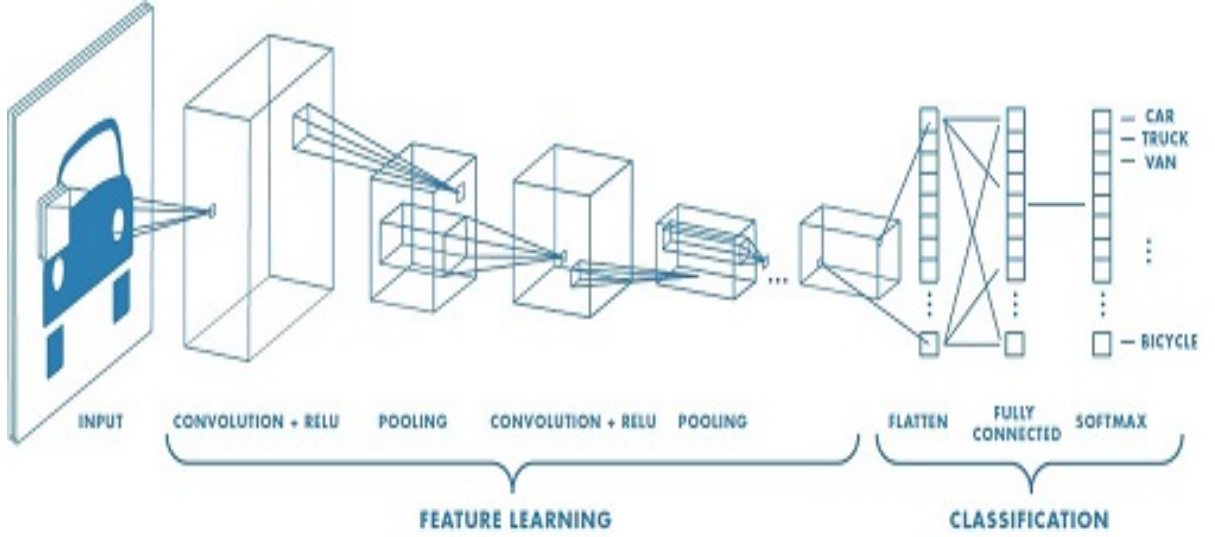


Figure 2.1: Convolution Neural Network

There is four main building block of CNN architecture :

- **Convolution** : Convolution process is normally defined as element-wise multiplication followed by addition. The purpose of Convolution operation is to extract unique features or pattern from the data matrix. Convolution operation preserves the unique relationship between data points by learning features using small squares of input which slide over data matrix, generally known as kernels (filter or feature detector), which produce output feature maps (convoluted feature or activation map). These filters slide over the data matrix to preserve the feature map. CNN learned these feature maps during the training phase using back-propagation. In CNN architecture, we need to specify the number of filters, filter size.

The size of feature maps(convoluted feature or activation map) depends upon three parameters which need to be decided by us:

- **Depth:** It represents the number of kernel, we are using to extract feature maps. Each kernel extracts different features from the same data matrix.

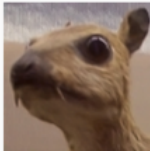

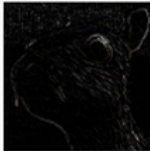

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

Figure 2.2: Filtes

- **Stride:** It is the number step by which the kernel slides over the data matrix. When we stride by one, it means we slide over a data matrix by one data point. If stride is two, then we stride by 2 data points over the data matrix. Higher the stride value is smaller the size of the resultant feature map.
- **Padding :** During stride, we lose the size of our data matrix to make the feature map equal to the data matrix, we add additional data points around the feature map. Zero padding inserts zero as a value in additional data points.
- **Non Linearity :** It is an point-wise operation (applied per data point of data matrix) that replaces all non-positive values in the resultant feature map by zero. The motive is to add non-linearity in our architecture since most of the real-world data we would want our architecture to learn would be non-linear. ReLU(Rectified linear unit), Sigmoid, Tanh function provide non-linearity of the network. Most

of the time, these are called activation functions. The output is also known as a rectified feature map when we use activation function as ReLU.

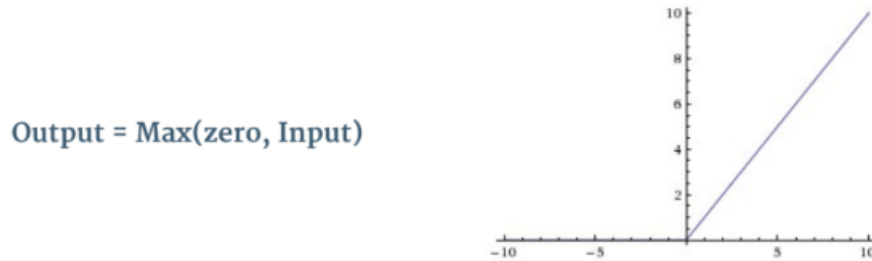


Figure 2.3: Rectified Linear Unit

- Max Pooling :** It means grouping together resources to maximize advantages. In-network it decreases the dimensionality of each resultant feature map but keeps the most meaningful informations. Most of the time, max-pooling is used for such purpose other than that there are many pooling techniques like average pooling, min pooling. It reduces the number of data points that reduce the computation overhead in the network and controls over-fitting(When a model remembers all data, a kind of error which we try to avoid).

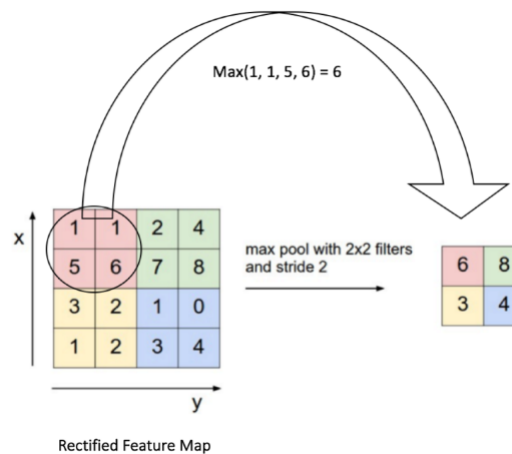


Figure 2.4: Max-pooling

- Fully Connected(Classification/regression) :** It can be viewed as a Multi layer Perception(MLP) that uses the soft-max activation function in the end layer of architecture for classification. There are various algorithms used in place of MLP

like Support Vector Machine(SVM) for regression used in finding boundary boxes around detected borders. It can be observed in Figure 4.4 at end of architecture.

Algorithm Approach of CNN2D

Data : $X_{i=0}^{n-1} = 2D \text{ data}, Y_{i=0}^{n-1} = Class_k$ //n = Total numbers of 2D data, k = Total class

Parameters : W_o, b_o, W_h, b_h // o,h = Parameter at output layer, hidden layer

Algorithm 2.1 CNN2D(X_i^n, Y_i^n)

- 1: Initialize parameter : Xavier Initializer
- 2: epochs $\leftarrow PositiveInteger$
- 3: while ($epoch \geq 0$) :
- 4: Convolve over data X_i using k*k kernel of depth d
- 5: maxpool on result of step 1 using m*m size with stride d'
- 6: Point wise ReLU operation as discussed chapter2.4
- 7: Flatten the resultant, Convert $2D \rightarrow 1D$
- 8: Compute Softmax for each data matrix X_i
- 9: Compute log_loss

$$loss = - \sum_{i=0}^{n-1} Y_i \log y'_i \quad (2.1)$$

- 10: Update parameters in backward direction using back-propagation
 - 11: epochs $\leftarrow epochs + 1$
-

Algorithm 2.2 Back-propagation

- 1: while(output_layer to Start_hidden_layer)
 - 2: for each neuron in selected layer
 - 3: Calculate SGD using chain rule and memorization
 - 4: Update parameters
-

Chapter 3

Speaker Recognition using CNN

We have discussed some of the text-independent methods in chapter 2.2, where the author had approached different methods for speaker recognition, like using a different audio feature or different Learning methods. In this chapter, we have proposed a Text-independent speaker recognition using CNN2D. We have focused on human cepstral based features which are theoretically capable of identifying speakers irrespective of their linguistics.

3.1 Data Description

LibriSpeech[4] is an ASR corpus of approximately ten speakers with a sampling rate at 16KHz read English speech, generated by Vassil Panayotov with the help of Daniel Povey.

3.2 System Flow

The proposed speaker Recognition can be seen in the below block diagram. It consists two steps : training step and testing step. In training step, one extra block is their training the data using convolution Neural Network which was absent from the test stage because we train our model on training data and test same model using unseen testing data or validation data.

- **Data Pre-Processing** :In this block, we prepare our data like cleaning the data, removing or identifying outliers, handling of zero value. It is a very crucial step in

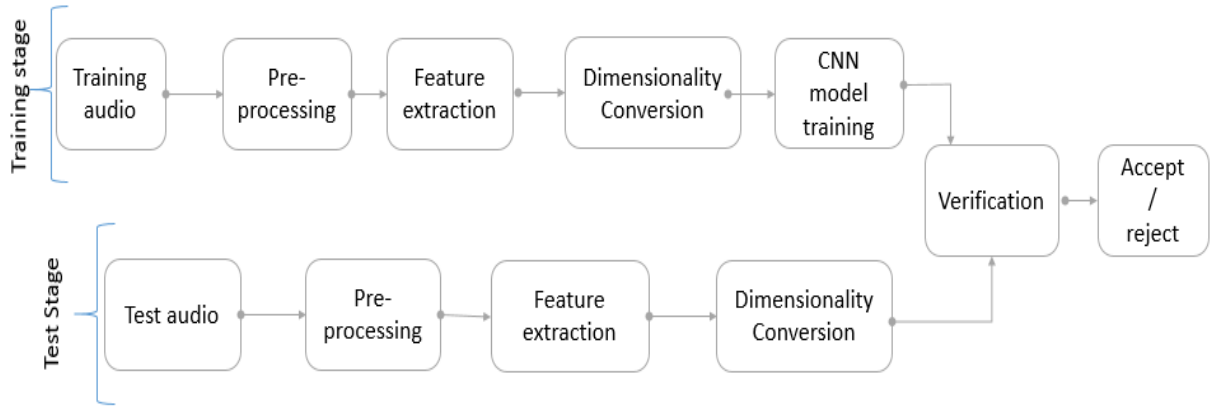


Figure 3.1: Proposed Speaker Recognition System

any system because if we have noisy data, then our model ends up learning noise, which degrades the result entirely.

- **Pre-Emphasis:** The voice produced by the human articular is in such a way that it reduces the energy of high frequency. Pre-emphasis is a boosting technique for higher frequency using a high pass filter, which makes frequency stronger and compensates for reduction caused by human articulators.

$$y(t) = x(t) - \alpha x(t - 1) \quad (3.1)$$

α = pre-emphasis factor lies between (0.9,1), where $x(t)$ is audio data point.

- **Framing:** It can be defined as a process of splitting audio data into a small size frame that can be overlapping. We need framing because as audio is a continuous sound, which means it changes very fast, looking for the pattern in continuous time series would be a loss. So we divide the audio into the small



Figure 3.2: Framing of Audio without overlapping

overlapping frames because, for that frame size, audio behaves like discrete data, and we can look for a pattern. Range from 15ms to 30ms frame size is supposed to be good because of such a small period of time audio data loss continuity.

- **Windowing:** It is a process of multiplying the audio frame by a time window of given shape to smoothen and make it less disturbed signal. To reduce discontinuity at the beginning and end of each frame, the signal must be reduced to zero or near to zero.

(i) Triangular window

$$w(n) = 1 - \left[1 - \frac{2n}{M-1}\right]; 0 \leq n \leq M-1 \quad (3.2)$$

(ii) Hamming window :

$$w(n) = 0.54 - 0.46 * \cos\left(\frac{2n\pi}{M-1}\right); 0 \leq n \leq M-1 \quad (3.3)$$

where M is length of window in samples.

- **Data splitting:** we split our data into training data and testing data or validation data with equal distribution according to classes. In most of the times we use 70:30 or 80:20 ratio for training data and test data.
- **Feature Extraction :** In this step, we extract cepstral features from each frame, the extraction process we have discussed in chapter 1.4. From each frame, cepstral features can be extracted from a range of 14 to 22. In this project work, we had used features Like MFCC, GFCC, and combine feature of MFCC & GFCC. We had detailed discussions in chapter 3.6 on which features had shown good results.

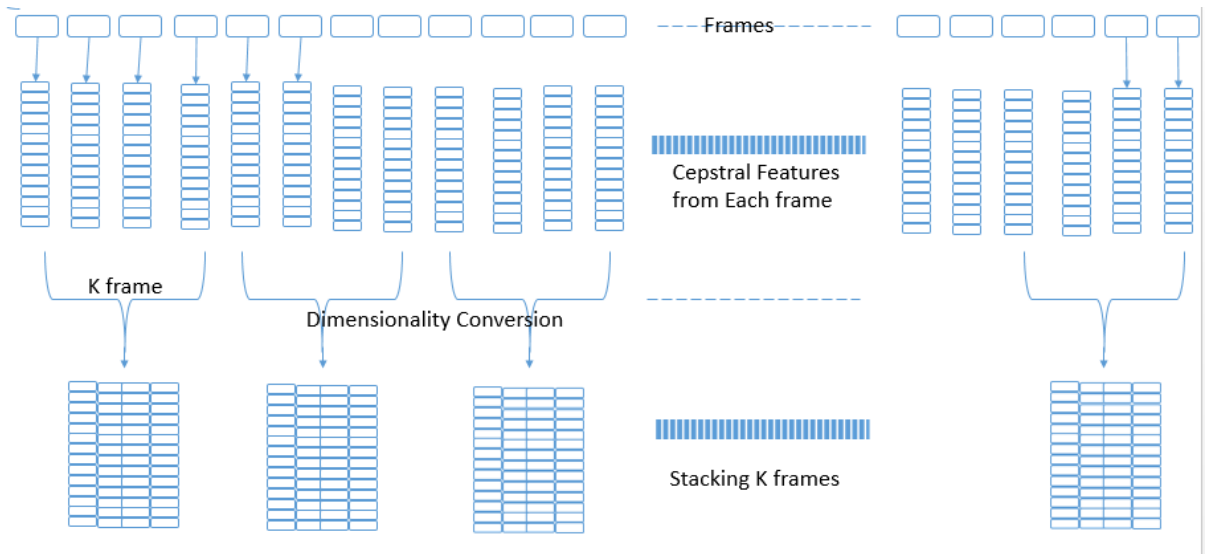


Figure 3.3: Process of conversion of 1D to 2D

- **Dimensionality conversion :** In this step, we transposed each frame and stacked horizontally each frame till the time period we want, like one second, two seconds according to our need as shown figure 3.3. Using this step, we converted our 1-Dimension data into 2-Dimensions. The motive of conversion was to split data based on time in the second through which we can observe a unique pattern for a time period.
- **Building Speaker Model :** We used the Convolution Neural Network 2D model for our speaker classification task. In our CNN model, we used one input layer, four convolution layer, two max-pooling layers, and two fully-connected layers.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 29, 38, 32)	320
conv2d_1 (Conv2D)	(None, 27, 36, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 13, 18, 32)	0
dropout (Dropout)	(None, 13, 18, 32)	0
conv2d_2 (Conv2D)	(None, 11, 16, 64)	18496
conv2d_3 (Conv2D)	(None, 9, 14, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 4, 7, 64)	0
dropout_1 (Dropout)	(None, 4, 7, 64)	0
flatten (Flatten)	(None, 1792)	0
dense (Dense)	(None, 256)	459008
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 10)	2570
Total params: 526,570		
Trainable params: 526,570		
Non-trainable params: 0		
None		

Figure 3.4: CNN Architecture for Speaker Recognition

Before reaching to above architecture we tuned different parameters to get a good architecture.

Hyper-parameters are :

- Epochs : 90 to 150 epoch is good for training our model. We have shown graph till 200 epoch to show convergence of our model.
 - Loss Function : we used loss function as Stochastic gradient descent(SGD).
 - Learning Rate : learning rate of 0.001 used with moment value of 0.9
 - Drop-Rate : After every max-pooling layer we used drop-rate of 0.5(probability).
- **Training model & Verification :** We trained our model on CNN2D architecture. We were suggested in fig architecture on the training data and did validation on training and test data.

3.3 Speaker Recognition Algorithm

Algorithm 1.1 Speaker Recognition

- 1: Data $\leftarrow S_i(k), Y_{label_i}$ // $S_i(k)$ is i^{th} audio data, Y_{label_i} label of i^{th} audio data
- 2: Split data into required audio time length

```
(15548, 1240) (15548,)
[[-1.87310592 -1.09493297  0.34074646 ... -0.0201444 -0.11725355
  -0.16679354]
 [-1.86579514 -1.14317253  0.58139276 ...  2.28693923  0.36326414
  -0.94033383]
 [ 0.93963426  0.90759552  0.51687359 ...  0.27535692 -0.40408093
  0.0175699 ]
 ...
 [-1.46864969  0.67282524  0.24423758 ...  0.04497515  0.2784393
  -0.05316988]
 [ 0.51655548 -3.81458742 -0.96105319 ... -0.0953755 -0.08106401
  0.03131777]
 [ 0.8837167 -0.0040365 -0.96126007 ...  0.03322116  0.17498973
  0.09935366]] ['1462' '1462' '1462' ... '422' '422' '422']
```

Figure 3.5: time domain audio data

- 3: for j in $S_i(k)$
- 4: Apply pre-emphasis // Discussed in chapter 3.1
- 5: $S'_i(k)$ is standardization of result we get from step 2 & 3

```

6: MFCC_ = MFCC( $S'_i(k)$ ).T // Discussed in chapter 1.4
7: GFCC_ = GFCC( $S'_i(k)$ ).T // Discussed in chapter 1.4
                                // data.T will transpose the data
8: Combined_feature = concatenate(MFCC_,GFCC_)
9:  $X_{data} \leftarrow$  Dimensionality conversion // Discussed in Chapter 3.2

```

```

Shape of Input data (15548, 31, 40, 1)
Training shape: (12438, 31, 40, 1), Training label shape : (12438, 10)
Training shape: (3110, 31, 40, 1), Training label shape : (3110, 10)
[[[[ 0.32704413]
    [-1.34916174]
    [ 0.70677   ]
    ...
    [-0.04610391]
    [-0.13420666]
    [-0.19524829]]

  [[-0.15077948]
    [-0.66748989]
    [ 0.21727663]
    ...
    [ 0.30838284]
    [ 0.02467821]
    [-0.20326397]]

  [[-0.2378814 ]
    [-0.71599686]
    [ 0.27907012]
    ...

```

Figure 3.6: 1D converted audio data into 2D

```

10:  $y_{predicted} \leftarrow$  CNN2D(  $X_{data}$ ,  $Y_{label_i}$ ) // discussed in chapter 2.4

```

3.4 Illustration

The setup used for experiments is on Google Colab[11], an online IDE with GPU provided by Google Inc. to students and researchers for experimenting with Machine Learning and Neural Networks.

The specification provided by Google Colab is as follows :

- CPU model name: Intel(R) Xeon(R) CPU @ 2.30GHz
- No. of Processors: 2
- CPU Cache Size: 46080 KB

- GPU Name: Tesla T4
- GPU Memory : 16 GB (15079MiB usable)
- RAM: 12.9 GB

The programming language used for conducting experiments in Python 3.6. The packages used for the experiment include SkLearn, Keras, Time, Tqdm, spafe, Librosa, SoundFile, NumPy, Matplotlib, etc.

3.5 Result

After tuning hyperparameter, we trained our model on training data which is 80% of audio data and tested on 20% of unseen data got the result as follows:

Training Accuracy:	96.01%
Test Accuracy:	97.46%
Log loss on Training:	0.0532
Log loss on Testing:	0.0761

Table 3.1: Speaker recognition result

In this result, we used a one-second audio sample, and from each frame, we extracted combine cepstral features [MFCC & GFCC].

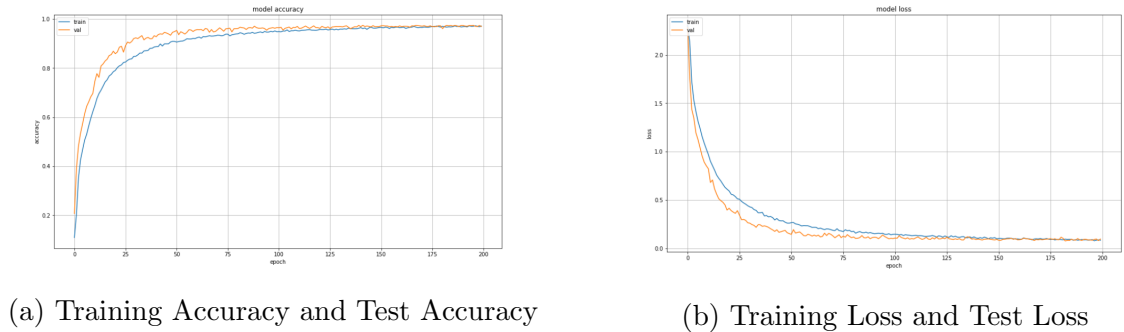


Figure 3.7: Performance Plot Accuracy & Loss

The accuracy plot 3.7a and confusion matrix 4.5 of test data show us that our model learned the pattern in speaker data and classify test data very accurately. But higher the accuracy there is the chance of overfitting(when there are low training error and difference between training error and

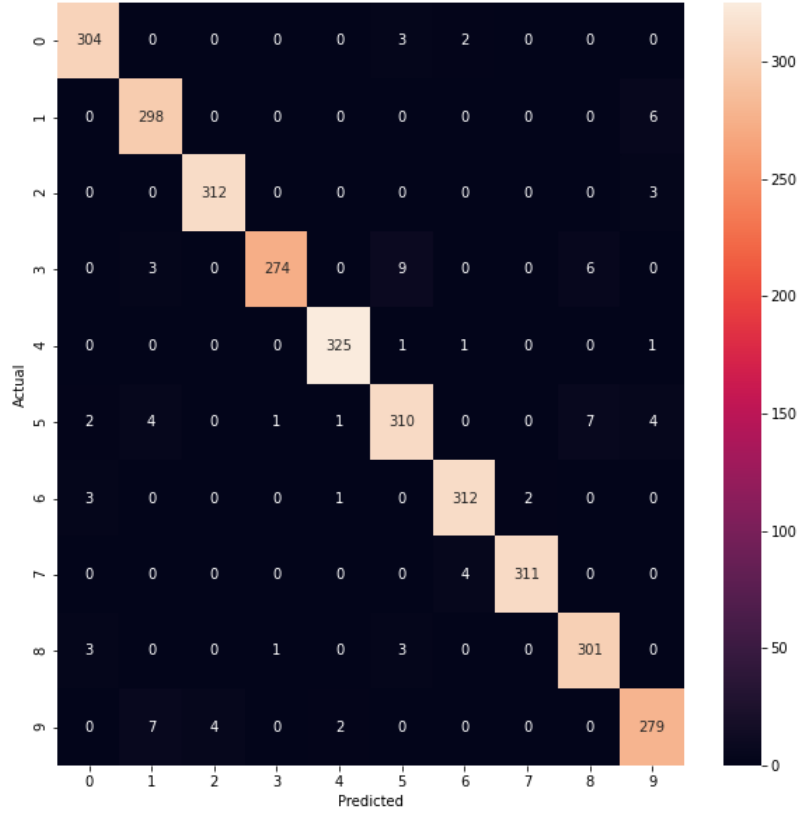


Figure 3.8: Confusion matrix of Speaker Classification of 10 user

test error are high) which doubt overshadowed from the loss plot 3.7b, we can observe that the bias and variance are well balanced

3.6 Conclusion

In this chapter, we addressed the problem of Speaker recognition using the CNN classification model. Not only we used CNN2D, but we also improved accuracy as compared to [1] and [6] using only one-second data while [1] and [6] used two-second to four-second audio data. More detailed observations were discussed in chapter 5.

Chapter 4

Speaker Liveliness Detection using Text-Independent Voice

4.1 Introduction

Voice biometric is easy to use, and its use in industry is increasing day by day, but anyone can have the recording of our voice data that can be used for presentation attack.

There are two types of presentation attacks, which is the most common regarding voice biometric:

- **Replay attack:** In this type of attack, impersonator uses the recorded sound of the speaker and tries to gain access by replaying the speaker's audio sample.
- **Mimicry Attack:** A professional actor tries to mimic the exact pattern of speaking as the speaker does and tries to break into the system.

So, detecting live speakers is also a necessary requirement in voice authentication or biometric authentication. In this chapter, we experimented with the liveliness of the speaker. In chapter 2.3, we discussed the different techniques of liveliness detection. [9] suggested that detection of liveliness based on Doppler shift: used in the Radar system but the experiment was based on the correlation of given audio sample vs enrolled audio sample, which means its Test-Dependent. Our goal was focused on Text-Independent speaker solution for ATMs and Mobile Phones.

Idea: When a user speaks, the system produces a continuous higher frequency which is unheardable to human ears and records both the user's voice and reflection produced by

human articulatory at high frequency, and later this reflection is observed for detecting liveliness. Loudspeaker which is used in smartphone to produce sound has a diaphragm which moves only forward direction and backward direction while the human sound production system articulator moves multidimensionally like up, down, back and forth. So from their properties, we can try to differentiate between simple movement and complex movement for liveliness.

Why higher frequency?

Doppler stated that,

$$\delta f \propto \frac{v \cos(\alpha)}{c} f_o \quad (4.1)$$

Where c = speed of sound in the medium,

v = articulatory movement,

α = angle between mouth and loudspeaker

f_o = High frequency

When we look at equation c, v and α is constant So,

$$\delta f \propto f_o \quad (4.2)$$

which means higher the frequency more change we can observe.

4.2 Data Collection

For liveliness, we collected our data. We used two devices which include one loudspeaker used for continuous production of a higher frequency of 15KHz in our experiment, and another device is the microphone which records reflection of higher frequency and the human voice, recorded at 32-bit with a 48KHz sampling rate. Distance between mouth and device is around 7cm to 10 cm.

For replay attract data, we did the same as above instead of a real human with replay record sound and then record it with high frequency at 32-bit with a 48kHz sampling rate.

4.3 Flow Diagram

The proposed Liveliness Detection can be seen in the below block diagram. We discuss Doppler shift, pre-processing and feature extraction, and liveliness model.

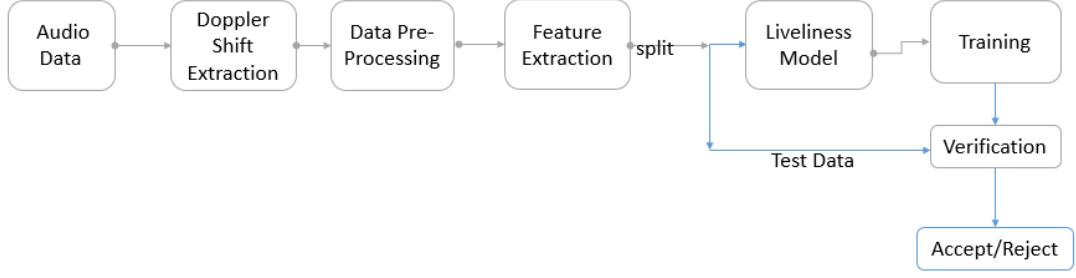


Figure 4.1: Block Diagram of Liveliness Detection

- **Doppler Shift Extraction**

To extract the Doppler shift from reflected high frequency, we used the Butterworth filter to cut the frequency of range from 14.5KHz to 15.5KHz for analysis for liveliness.

- **Data pre-processing** In preprocessing, we remove silence and white noise from audio using the Audacity framework as well as frame the data with 30ms of window length.

- **Feature Extraction**

We extracted spectral entropy [9].

Steps for calculation of Spectral Entropy :

Algorithm 4.1 Spectral Energy

1. We divide the frame in L equal fragment(sub-bands) each with W_L length.
2. Calculate the Energy for every sub-band

$$E_{subFrame}(i) = \frac{1}{W_L} \sum_{n=1}^{W_L} |x(i)|^2 \quad (4.3)$$

where, $x(i)$ is audio data point, W_L is length of sub-band

3. Calculate the sum of energy of every sub-band

$$E_{shortFrame_i} = \sum_{K=1}^L E_{subFrame_k} \quad (4.4)$$

$E_{shortFrame_i}$ is total energy of a frame, $E_{subFrame_k}$ is energy of k^{th} sub-band.

4. Calculate sequence of probability for every sub-band

$$e_f = \frac{E_{subFrame_f}}{E_{shortFrame_i}} \quad (4.5)$$

where e_f is sequence probability of subframe, $1 \leq f \leq W_L$

5. calculate Entropy of each short frame

$$H(i) = - \sum_{f=1}^{W_L} e_f \cdot \log_2(e_f) \quad (4.6)$$

where $1 \leq H(i) \leq L$

6. Repeat steps 1 to 5 for each frame.

- **Liveliness Model** For liveliness classification problem we used Artificial Neural Network with two fully connected layer; have 128 neurons at each layer. As regularization method we used Dropout with probability of 0.6 and we used Stochastic Gradient Descent(SGD) as optimization function.

Model: "sequential_7"

Layer (type)	Output Shape	Param #
dense_24 (Dense)	(None, 128)	240768
dropout_17 (Dropout)	(None, 128)	0
dense_25 (Dense)	(None, 128)	16512
dropout_18 (Dropout)	(None, 128)	0
dense_26 (Dense)	(None, 2)	258
Total params: 257,538		
Trainable params: 257,538		
Non-trainable params: 0		

Figure 4.2: ANN architecture for Liveliness Detection

Before reaching to above architecture we tuned different parameters to get a good architecture.

Hyper-parameters are :

- Epochs : 90 to 120 epoch is good for training our model. We have shown graph till 200 epoch to show convergence of our model.
 - Loss Function : we used loss function as Stochastic gradient descent(SGD).
 - Learning Rate : learning rate of 0.001 used with moment value of 0.9
 - Drop-Rate : After every dense layer we used drop-rate of 0.6(probability).
- **Verification** For verification, we split the 20 percent of data for testing which is unseen to model. And for each data point, we calculated soft-max function, which calculates class probability. The class which as height probability data labeled as that class.

$$Softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^k e^{z_j}} \quad (4.7)$$

where z_i is class probability of class (i) and k is total class.

In case of Binary class classification k will be two.

4.4 Liveliness Detection Algorithm

Data : $X_{i=0}^{n-1} = 1D \text{ data}, Y_{i=0}^{n-1} = Class_k$ //n = Total numbers of 1D data, k = Total class

Parameters : W_o, b_o, W_h, b_h // o,h = Parameter at output layer, hidden layer

```
[[ -1.83640816 -0.22880208  0.20060539 ... -1.1899315  -2.77224288
   1.38523936]
 [  0.46987228 -0.34835952 -0.74741294 ...  2.85600518 -0.8540223
  -0.12901431]
 [  0.75071371 -0.01855648  0.1823154  ... -1.14288062 -0.67584429
  -0.14175205]
 ...
 [-0.15697777 -0.1752219  -0.07329063 ... -1.10299738  1.05128033
  -0.26184916]
 [-0.39433618  0.43522624  0.28592332 ...  0.70032342 -0.67211928
  -0.6548004 ]
 [  0.13178664  0.20272787 -0.48502296 ... -1.55626776  1.68367822
   0.58848686]]
(5976, 1880) (5976,)
Shape of Input data (5976, 1880)
```

Figure 4.3: Input data for liveliness detection

Algorithm 4.2 Liveliness Detection

- 1: $S'(K) \leftarrow$ Extract Doppler shift from audio data in range of 14.515.5 KHz from audio data $S(K)$ //discussed in chapter 4.2
 - 2: Remove silence noise
 - 3: Standardized the result of step 2
 - 4: Frame the audio data $S'(k)$ into m equal frame with 60% overlapping
 - 5: for $frame_{i=0}$ to m
 - 6: extract spectral entropy // discussed in chapter 4.3
 - 7: $audio_data_i \leftarrow$ Append spectral entropy of 1sec audio data in 1D
 - 8: $y_{predicted} \leftarrow ANN(audio_data_i, y_{label_i})$
-

Algorithm Approach of ANN

Algorithm 4.3 $ANN(X_i^n, Y_i^n)$

- 1: Initialize parameter : Xavier Initializer
 - 2: $epochs \leftarrow PositiveInteger$
 - 3: while ($epoch \geq 0$) :
 - 4: propagate forward in network and mutiply with parameter with input at each layer
 - 5: Compute activation function as output of each neuron
 - 6: Compute log_loss
 - 7: compute gradient discent w.r.t loss
 - 10: Update parameters in backward direction using back-propagation
 - 11: $epochs \leftarrow epochs + 1$
-

$$loss = - \sum_{i=0}^{n-1} Y_i \log y'_i \quad (4.8)$$

Algorithm 4.4 Back-propagation

- 1: while(output_layer to Start_hidden_layer)
 - 2: for each neuron in selected layer
 - 3: Calculate SGD using chain rule and memorization
 - 4: Update parameters
-

4.5 Result

After tuning hyperparameter, we trained our model on training data which is 80% of audio data and tested on 20% of unseen data got the result as follows:

Log loss on Training:	0.432
Log loss on Testing:	0.261
sensitivity of LiveUser	96.90%

Table 4.1: Speaker recognition result

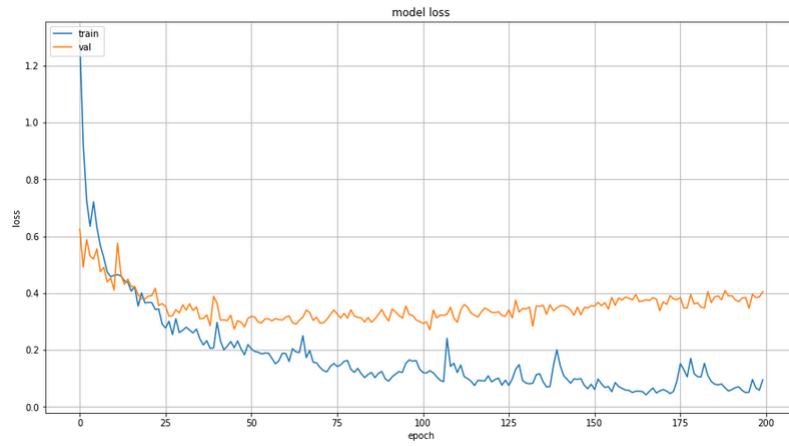


Figure 4.4: Loss plot Liveliness Model

As from the confusion matrix and loss plot, we can observe that the model is tried to learn patterns on data in which performance accuracy we achieved is 89% on the text-independent model. The difference between training loss and training is slightly higher but still had a very small difference which suggests the model is neither over-fitting nor under-fitting.

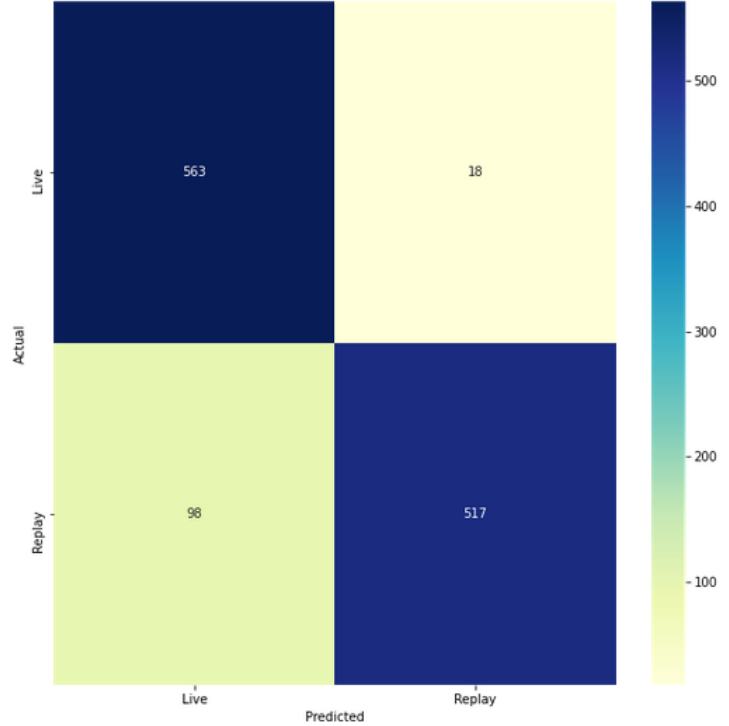


Figure 4.5: Confusion matrix of Liveliness model

The reason for low performance could be that both type of data had one common 15KHz frequency while the shift is different but very less which could be the reason for affecting our result. Despite the commonness, the sensitivity of Live user detection is higher at around 97% it means only 3% percent of live users classified as a replay attack.

4.6 Conclusion

In this chapter, we have addressed the problem of text-independent liveliness detection using an Artificial Neural Network for live user detection and we achieved a sensitivity of liveliness around 97%.

Chapter 5

Comparative Analysis of Various Text-Independent Voice Recognition Model

5.1 Introduction

In chapter 3, we have used CNN for Text-Independent Speaker Recognition, which gave us a good performance of the model on LibriSpeech data. In this chapter, we have used different models like Artificial Neural Network (ANN,) Support Vector Machine (SVM) as a base model to compare performance with our proposed architecture.

Our aim was to introduce voice Biometric in ATMs and Mobile Phones for banking transactions. So our focus was on improving accuracy with less data which we will observe in Chapter 5.2 apart from that in Chapter 3.4, we achieved an accuracy of around 98% with one audio sec data.

5.2 Comparative Analysis

5.2.1 Performance of Models for different Audio Length

In this section, we checked for the performance of the model over a different feature like with MFCC, GFCC, and combined cepstral feature [MFCC, GFCC] over different time periods and different models like CNN2D, ANN, and SVM. We are extracting 20 cepstral features from each frame [6].

- CNN2D System Performance Observed over Different Audio Length(sec)

CNN2D System Performance Observed over Different Audio Length(sec)			
time(sec)	MFCC	GFCC	MFCC+GFCC
1	89.91%	95.88%	97.46%
2	96.67%	98.44%	97.99%
3	97.35%	98.43%	96.76%
4	97.77%	98.42%	98.08%
5	95.19%	96.67%	97.17%

Table 5.1: CNN2D System Performance vs Audio Length

CNN2D with cepstral feature MFCC can be observed that performance improved for two-second to four-second audio data. Three sec and four-second audio data had approx the same performance accuracy of 97%(approx). CNN2D with cepstral feature GFCC performance also follows the same trail as CNN2D with cepstral feature MFCC but with an improved accuracy difference of around (1,2)% which is approximately equal to an accuracy of 98%. CNN2D with a combination of MFCC & GFCC cepstral features gave us good performance over one-second audio data. It gave good accuracy of 98% at four-second audio data.

- ANN's Performance observed under different Audio length(sec)

In the case of ANN with the cepstral feature, MFCC performed poorly over the different audio length data, and we can observe that it is decreasing, it achieved

ANN's Performance observed under different Audio length(sec)			
time(sec)	MFCC	GFCC	MFCC+GFCC
1	81.28%	90.95%	95.19%
2	77.13 %	81.85%	87.74%
3	73.35%	77.6%	80.01%
4	68.77%	71.63%	79.0%
5	60.03%	62.75%	70.54%

Table 5.2: ANN System Performance vs Audio Length

the highest performance at around one-second audio data the same trail followed by ANN with cepstral feature GFCC, ANN with combined cepstral features[MFCC, GFCC]. ANN with combined cepstral features gave the highest performance accuracy of around 95.16

- SVM System Performance Observed over Different Audio Length(sec)

SVM System Performance Observed over Different Audio Length(sec)			
time(sec)	MFCC	GFCC	MFCC+GFCC
1	87.03 %	88.03 %	89.61 %
2	79.10 %	79.88 %	79.55 %
3	75.40 %	76.18 %	75.60 %
4	72.16 %	74.95 %	75.08 %
5	73.26 %	74.23 %	74.82 %

Table 5.3: SVM System Performance vs Audio Length

SVM with one-second audio data outperforms the other audio length data. SVM with combined cepstral features gave the highest performance, An accuracy of around 89% while the increase in audio length performance of MFCC, GFCC, and combined cepstral feature decreased.

There could be two reasons for degrading performance over the increase in the length of audio data in sec: one can be with the increase in audio length feature vector increase, and another one could be with the increase in audio data in sec training data size decreased.

With the use of the Dimension reduction algorithm, the feature vector can be reduced, which may help in an increase in performance and using frame overlapping, we can generate more audio data.

5.2.2 Performance Analysis based on Different ratio of overlapping frames

Overlapping frame means one rolling window over rolling another, for e.g., 50 percent overlapping means the first window's last 50 percent data will be in the second window's first 50 percent.

So in this section, we observe the performance of models over a different percentage of overlapping frames. Now we had data from chapter 5.2.1 that models with a combined cepstral feature[MFCC, GFCC] had a good performance over one-second audio data. We are using the same parameters now we observe the change in performance over the overlapping frame.

Performance observed under Different Overlapping Frames			
Frame Overlapping	CNN2D	ANN	SVM
50 %	96.61 %	80.06 %	58.39%
60 %	97.68 %	88.60 %	76.50%
70 %	98.23 %	92.39 %	81.32%
80 %	98.01 %	95.16 %	89.61%

Table 5.4: Performance based on different Overlapping Frame Ratio

CNN2D with the combined cepstral feature over the different percentages of overlapping; increases the performance of the system. But with 70 percent and 80 percent,

performance improvement is very minute. ANN with the combined cepstral feature had increased in performance from 50% frame overlapping to 80% frame overlapping with an accuracy increased from 80% to 95.6%. SVM with the combined feature shows the increase in performance accuracy over the increase in the overlapping frame in percentage. From 59% accuracy with 50% frame overlapping performance increased to an accuracy of around 89.61% with an 80% frame overlapping.

5.2.3 Performance observation under different Number of Speaker

As each step, we are taking the best parameters from the previous experiment. In this section, we are going to observe the performance of the different models over various numbers of users. For CNN2D we will use 60% overlapping because it had the same performance as it had with 70% and 80% approximately. For ANN and SVM, we will use an 80% frame overlapping as we observed in chapter 5.2.2. we observed in chapter 5.2.2 with audio data size is one sec and extracted 20 cepstral features from each frame had a good performance.

Performance observed under Different Number of Speakers								
No. of speaker	2	3	4	5	6	7	8	9
CNN(%)	99.35	99.68	98.46	98.25	96.60	97.66	97.23	97.29
ANN(%)	99.28	98.61	98.30	97.42	96.01	95.72	95.93	95.72
SVM(%)	97.13	97.88	96.02	94.56	92.44	90.16	90.10	89.88

Table 5.5: Performance observed under Different Number of Speakers

The performance of each model increases when the number of users is less. Two speakers are doing relatively well. With fewer speakers, training epochs also decreased in the learning process as well as training time.

CNN2D from speakers 2 to 4 needed only 30 epoch to fully converge while other speakers needed more than 30 epochs for learning. ANN also follow the same trail as CNN but needed more epochs than CNN architecture. In SVM performance increased with a

decrease in the number of speakers. In this experiment, our proposed method is outperforming other models in every aspect, and our model required fewer epochs and fewer audio data to learn.

For more confusion matrix data look at **Appendix A**.

5.3 Observation and Conclusion

In this chapter, we have discussed the proposed model's performance over the various parameters as well as other base models. Our proposed solution outperformed other model's performance in every aspect.

From our observation, we concluded that CNN2D with a combined cepstral feature with frame overlapping of around 60% with one-sec audio data, and from each frame 20 cepstral features extracted gave us a good performance.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In chapter 3, we have discussed speaker recognition architecture. we observed that with combined cepstral features we can achieve good performance over single cepstral features. Later in chapter 5, we had a wide range of experiments where we check our proposed model with base models that over-perform other base models.

From chapter 5, we got various results of our model which achieved high performance with one-second audio data. Proposed model with an increase in the number of the speakers the performance differed only by a fraction, where other base models had observed a sharp decrease in performance. Also, our model achieved high performance over 60 % frame overlapping while the base model needs 80% to 90% of frame overlapping to achieve model accuracy which is still less than our proposed method. At last, the training time of the purposed method is less very compared to other models.

In chapter 4, we differentiate live user and replay attack with 90% accuracy over unseen test data using the text-independent audio sample.

6.2 Future Work

We succeeded in our aim for speaker classification with one-second audio data but in our speaker recognition data had 10-user with around 20min-25min audio data each user. In the next step of this project, I would like to work on a decrease in the size of the training data. As well as text-independent liveliness detection needed more work like increasing

performance.

Making this project into the product will be the result of my future work.

References

- [1] Khan Suhail Ahmad et al. “A unique approach in text independent speaker recognition using MFCC feature sets and probabilistic neural network”. In: *2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR)*. IEEE. 2015 JAN,4, pp. 1–6.
- [2] Bakhtiar Affendi Rosdi, Chai Wuh Shing, and Shahrel Azmin Suandi. “Finger vein recognition using local line binary pattern”. In: *Sensors* 11.12 (2011), pp. 11357–11371.
- [3] Theodoros Giannakopoulos and Aggelos Pikrakis. *Introduction to audio analysis: a MATLAB® approach*. Academic Press, 2014.
- [4] Vassil Panayotov et al. “Librispeech: an asr corpus based on public domain audio books”. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2015, pp. 5206–5210.
- [5] Linghan Zhang et al. “VoiceLive: A Phoneme Localization Based Liveness Detection for Voice Authentication on Smartphones”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS ’16. Vienna, Austria: Association for Computing Machinery, 2016, pp. 1080–1091. ISBN: 9781450341394. DOI: 10.1145/2976749.2978296. URL: <https://doi.org/10.1145/2976749.2978296>.
- [6] Andrew Boles and Paul Rad. “Voice biometrics: Deep learning-based voiceprint authentication system”. In: *2017 12th System of Systems Engineering Conference (SoSE)*. IEEE. 2017, pp. 1–6.
- [7] Huan Feng, Kassem Fawaz, and Kang G Shin. “Continuous authentication for voice assistants”. In: *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. 2017, pp. 343–355.

- [8] Medikonda Jeevan et al. “Robust speaker verification using GFCC based i-Vectors”. In: *Proceedings of the International Conference on Signal, Networks, Computing, and Systems*. Springer. 2017, pp. 85–91.
- [9] Linghan Zhang, Sheng Tan, and Jie Yang. “Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 2017, pp. 57–71.
- [10] Qian Wang et al. “Voicepop: A pop noise based anti-spoofing system for voice authentication on smartphones”. In: *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE. 2019, pp. 2062–2070.
- [11] *A cloud based data science work space similar to jupyter notebook*. <https://colab.research.google.com>.
- [12] *Unique mobile features designed to make it quick and easy for you to sign on, manage your accounts and perform transactions from your smartphone*. <https://www.wellsfargo.com/com/ceo/ceo-mobile/>.

Appendix A

Appendix

Performance observed under Different Number of Speakers :

A.1 2 Speaker Confusion Matrix

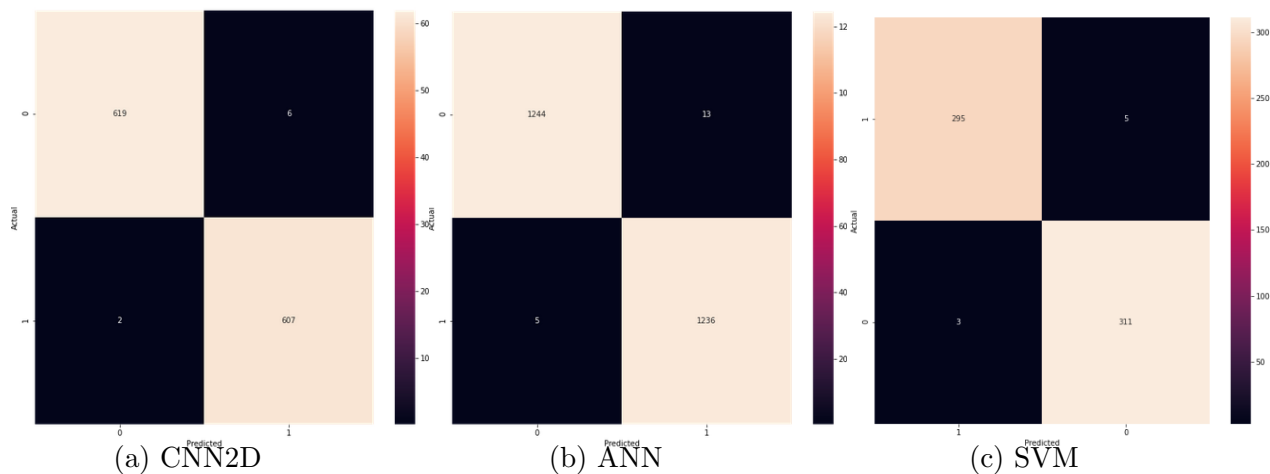


Figure A.1: Confusion Matrix of 2 Speaker

A.2 3 Speaker Confusion Matrix

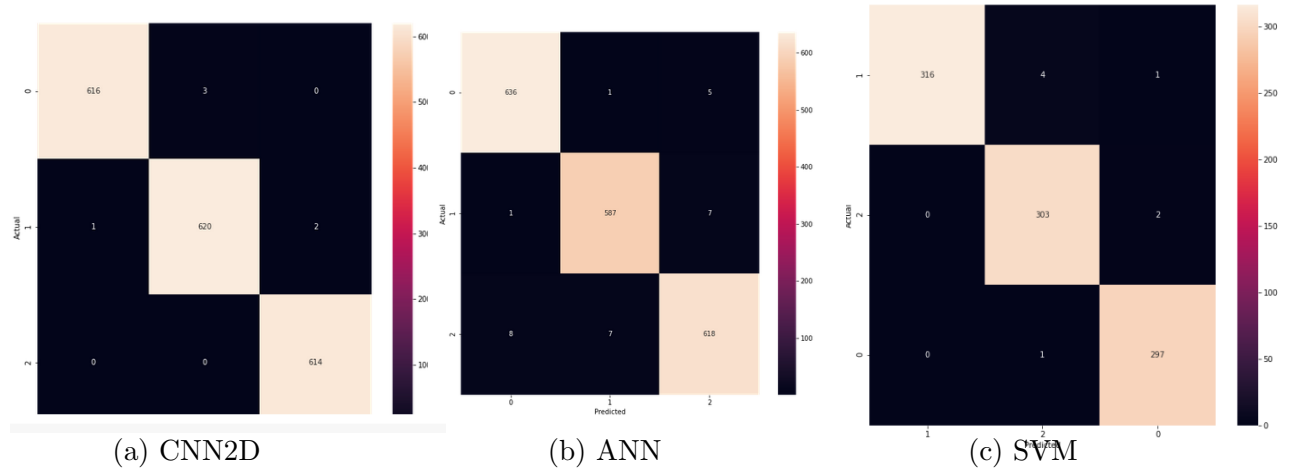


Figure A.2: Confusion Matrix of 3 Speaker

A.3 4 Speaker Confusion Matrix

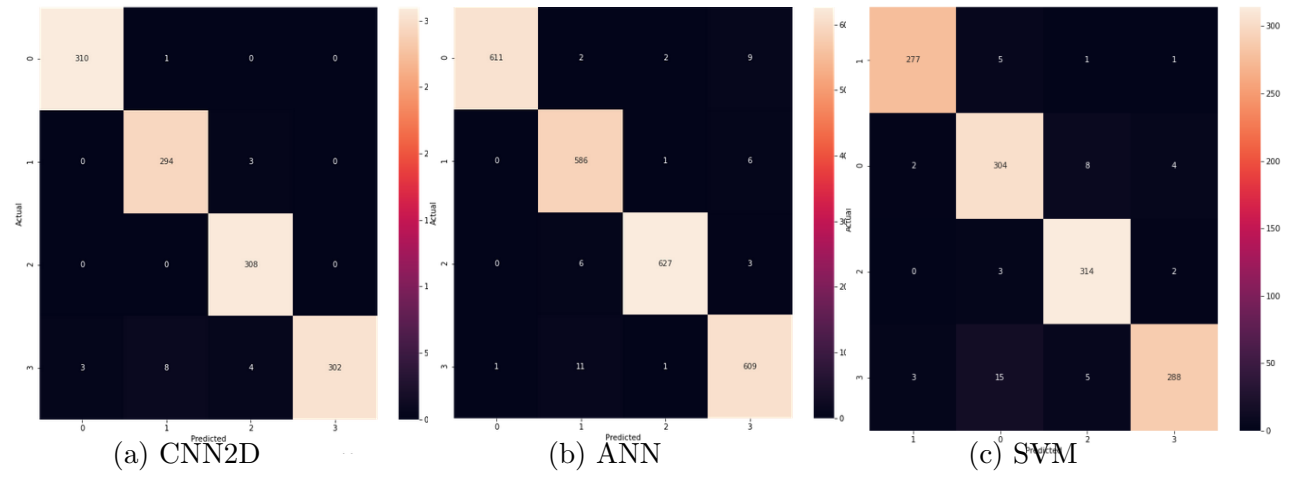


Figure A.3: Confusion Matrix of 4 Speaker

A.4 5 Speaker Confusion Matrix

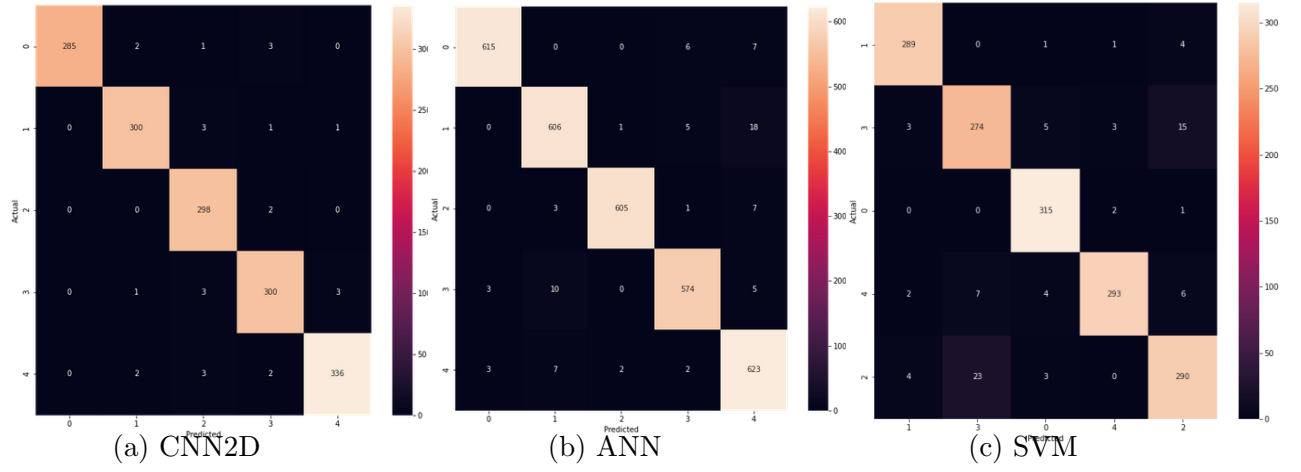


Figure A.4: Confusion Matrix of 5 Speaker

A.5 6 Speaker Confusion Matrix

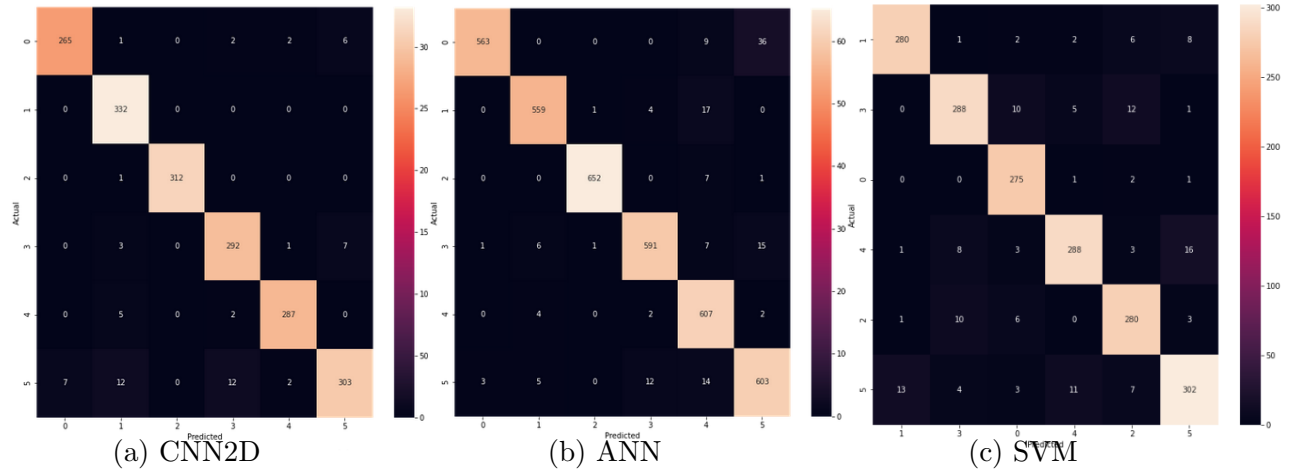


Figure A.5: Confusion Matrix of 6 Speaker

A.6 7 Speaker Confusion Matrix

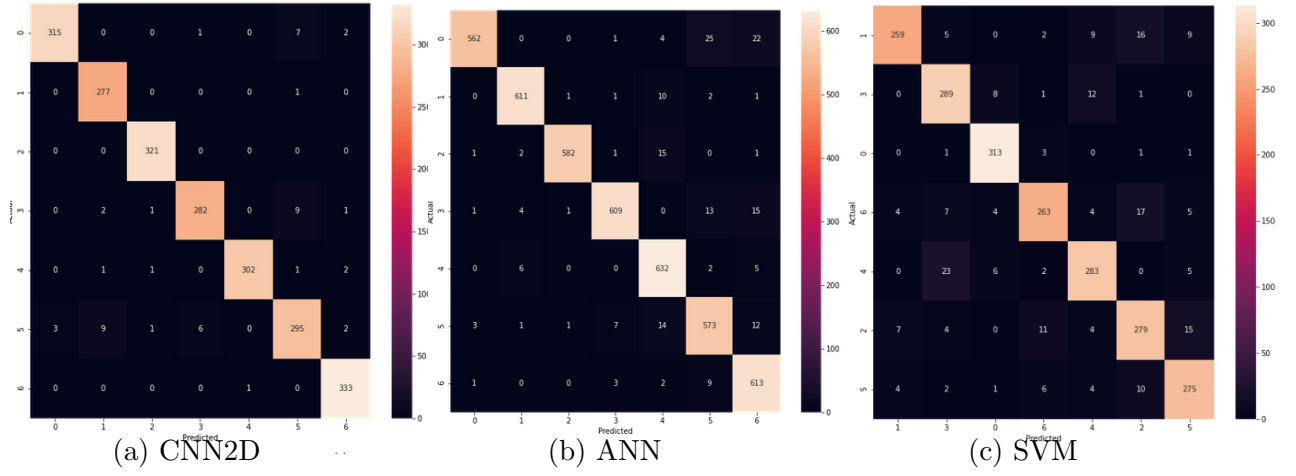


Figure A.6: Confusion Matrix of 7 Speaker

A.7 8 Speaker Confusion Matrix

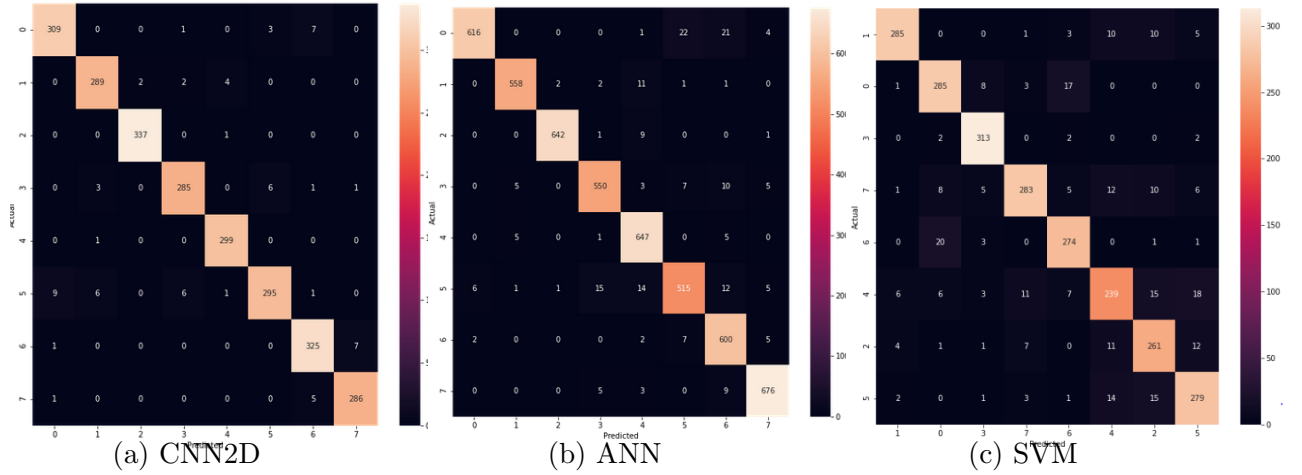


Figure A.7: Confusion Matrix of 8 Speaker

A.8 9 Speaker Confusion Matrix

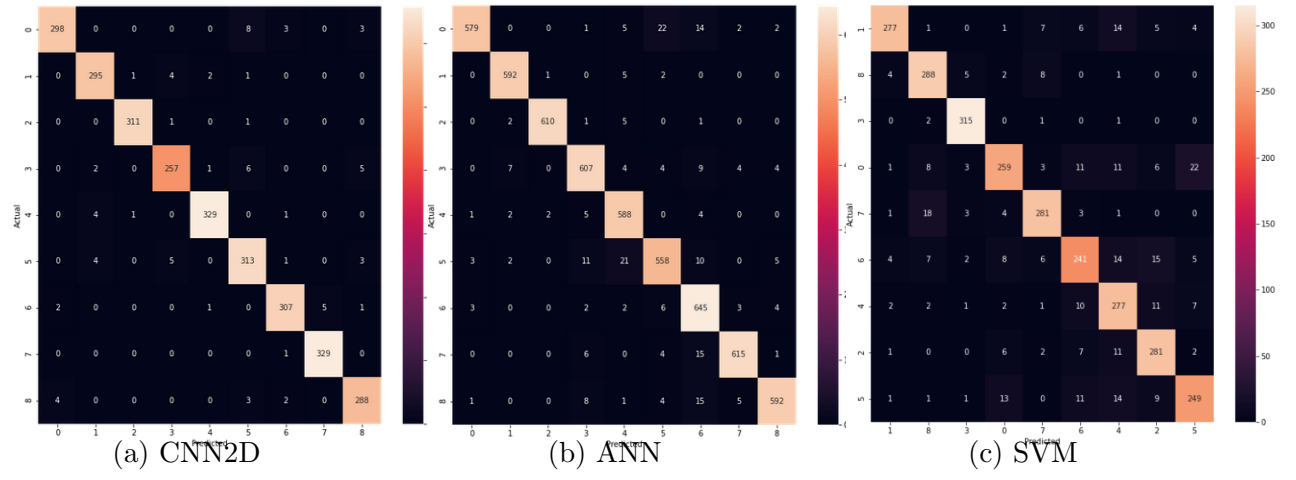


Figure A.8: Confusion Matrix of 9 Speaker

Speaker Recognition and Detection of Liveliness of Voice in ATMs & Mobile Phones

by Ritik Raj

Submission date: 29-Jun-2020 02:10PM (UTC+0530)

Submission ID: 1351255787

File name: ognition_and_Detection_of_Liveliness_of_Voice_in_ATMs_Mobile.pdf (1.48M)

Word count: 7548

Character count: 37845

Speaker Recognition and Detection of Liveliness of Voice in ATMs & Mobile Phones

ORIGINALITY REPORT

3%

SIMILARITY INDEX

0%

INTERNET SOURCES

1%

PUBLICATIONS

3%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to University of Hyderabad,
Hyderabad

Student Paper

1%

2

"Proceedings of the International Conference on
Signal, Networks, Computing, and Systems",
Springer Science and Business Media LLC,
2017

Publication

<1%

3

Thiramdas Narendra, Athulya M.S., Sathidevi
P.S.. "Classification of Pitch Disguise Level with
Artificial Neural Networks", 2019 International
Conference on Communication and Signal
Processing (ICCSP), 2019

Publication

<1%

4

Linghan Zhang, Sheng Tan, Jie Yang, Yingying
Chen. "VoiceLive", Proceedings of the 2016
ACM SIGSAC Conference on Computer and
Communications Security - CCS'16, 2016

Publication

<1%

5	Submitted to Pearson College Student Paper	<1 %
6	diginole.lib.fsu.edu Internet Source	<1 %
7	www.maxwellsci.com Internet Source	<1 %
8	Submitted to University of Wales central institutions Student Paper	<1 %
9	Submitted to UT, Dallas Student Paper	<1 %

Exclude quotes On

Exclude matches < 14 words

Exclude bibliography On