SPEAKER RECOGNITION BASED BIO-METRIC SYSTEM FOR INDOOR APPLICATION

PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF **BACHELOR OF ENGINEERING** IN **COMPUTER SCIENCE AND ENGINEERING** OF THE ANNA UNIVERSITY

PROJECT WORK

Submitted by

ARRASURA PARAMESH
1917107

JEYAPRATHAP S
1917123

RAHUL R
1917138

SRIRAM S
1917147

2023

Under the Guidance of
Dr.R.MUTHURAM M.E, Ph.D.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT COLLEGE OF TECHNOLOGY
(An Autonomous Institution affiliated to Anna University)

COIMBATORE – 641 013

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University)

COIMBATORE – 641 013

PROJECT WORK

APRIL 2023

This is to certify that this project work entitled

SPEAKER RECOGNITION BASED BIO-METRIC SYSTEM FOR INDOOR APPLICATION

Is the bonafide record of project work done by

ARRASURA PARAMESH

1917107

JEYAPRATHAP S

1917123

RAHUL R

1917138

SRIRAM S

1917147

of B.E (Computer Science and Engineering) during the year 2022 – 2023

______________________________	______________________________
Project Guide	Head of the Department
Dr.R.MUTHURAM M.E, Ph.D.	**Dr.J.C.MIRACLIN JOYCE PAMILA M.E, PhD.**

Submitted for the Project Viva-Voce examination held on ____________

____________________________	____________________________
Internal Examiner	External Examiner

ACKNOWLEDGEMENT

ACKNOWLEDGMENT

We express our sincere gratitude to **Dr.K. MANONMANI**, Ph.D., Principal, Government College of Technology, Coimbatore for providing us all facilitiesthat we needed for the completion of this project.

We whole-heartedly express our thankfulness and gratitude to **Dr.J.C.MIRACLIN JOYCE PAMILA**, M.E., Ph.D., Professor and Head of the Department, Computer Science and Engineering, Government College of Technology, for helping us to successfully carry out this project**.**

Our thankfulness and gratitude to our respectable project guide **Dr. R. MUTHURAM** M.E. Ph.D., Assistant Professor, Department of ComputerScience and Engineering who has been an immense help through the various phases of the project. With her potent ideas and excellent guidance, we were able to comprehend the essential aspects involved.

We extend our sincere thanks to **Dr.S. RATHI**, M.E., Ph.D., Professor (CAS), **Dr.K. KUMAR**, M.E., Ph.D., Associate Professor (CAS), **Dr.T. RAJA SENBAGAM**, M.E., Ph.D., Assistant Professor, **Dr.A. MEENA KOWSHALYA,** M.E., Ph.D., Assistant Professor, **Dr.R. BHAVANI**, M.E., Ph.D., Assistant Professor, **Prof.L. SUMATHI**, M.E, Assistant Professor for all their valuable suggestions to the completion of the project.

We thank all the non-teaching staff and our friends for their cooperation towards the successful completion of the project. We would like to dedicate the work to our parents for their constant encouragement throughout the project.

SYNOPSIS

SYNOPSYS

Speaker recognition aims to identify and verify the identity of a person based on their voice. Traditional speaker recognition systems have used various signal processing techniques and statistical modeling approaches. However, Deep learning has emerged as a powerful tool for speaker recognition, offering improved accuracy and robustness compared to traditional machine learning approaches.

A speaker recognition system using deep learning typically consists of several components, including a feature extractor, a ConvNet, and a decision-making module. The feature extractor is responsible for extracting relevant features from the audio signal, such as Mel frequency cepstral coefficients (MFCCs) or filter bank energies. These features are then fed into a neural network, which learns to classify them into different speaker identities.

The proposed system implements a convolutional neural network which is trained to extract the temporal pattern present in the audio signal. The CNN is trained with 10 different speaker classes having 4000 samples in each class.

During inference, the model takes a new audio sample as input and extracts its features using the feature extractor which is MFCC coefficients. The SoftMax at last layer of CNN assigns a probability distribution over the possible speaker identities, and the class with higher probability will be given as output.

The system also contains a Noise reduction module which filters out noise present in the audio signal. The reconstructed noise is then given to the speaker recognition model for classification. The proposed system is robust to noise and gives good accuracy during the real time implementation. The system is highly scalable and can be implemented in larger scale without any modification.

TABLE OF CONTENTS

TABLE OF CONTENTS

Government College of Technology Coimbatore - 641 013

Government College of Technology Coimbatore - 641 013

LIST OF FIGURES

Government College of Technology Coimbatore - 641 013

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Speaker recognition based biometric system uses a person's voice pattern as a unique identifier. Convolutional Neural Network has been promising in finding temporal features and model complex relationships between input data and speaker identities.

A CNN-based speaker recognition system typically consists of two main components: feature extraction and speaker classification. In the feature extraction stage, the input speech signal is transformed into a spectrogram, which is a visual representation of the signal's frequency content over time. The spectrogram is then processed through a CNN, which learns to extract relevant features from the signal. Training a CNN-based speaker recognition system involves feeding the network with a large dataset of speech signals and their corresponding speaker identities. The network learns to extract relevant features from the signals and map them to the correct speaker identities using a loss function, such as cross-entropy loss. CNN have shown promising results in improving the accuracy and robustness of speaker recognition.

1.2 PROBLEM STATEMENT

Speaker recognition based biometric system that identifies individuals based on their unique voice patterns. Traditional speaker recognition systems have used various signal processing techniques and statistical modelling approaches. However, these systems often suffer from low accuracy and poor performance in real-world scenarios due to several challenges.

One of the main challenges in speaker recognition is the variation in speech signals caused by different speaking styles, accents, and background noise. These variations can significantly affect the accuracy of speaker recognition systems, making it difficult to distinguish between different speakers.

Furthermore, traditional speaker recognition systems typically rely on hand-engineered features that may not capture all the relevant information in speech signals. This can lead to poor performance in complex scenarios, such as identifying speakers with similar voices.

But CNN can learn discriminative features from raw speech signals and model complex relationships between input data and speaker identities.

Taking these factors into consideration, we are proposing the 3 layered architecture for speaker recognition system with custom spectral gating-based noise reduction Mechanism. The main contribution of this document are as follows:

- 3 layered architecture has been proposed for speaker recognition system which includes edge layer, Database layer and web application layer for visualizing database.

- We are introducing a spectral gating-based noise reduction method which involves computing a spectrogram of a signal and estimating a noise threshold for each frequency band of that signal. That threshold is used to compute a mask, which gates noise below the frequency-varying threshold.

- We are also proposing a database layer which has MySQL database running on Apache, that receives inference from the edge layer and update the same in database.

- Based on the results of the experiments, 3 layers architecture for speaker recognition system has been proposed.

1.3 DATASET

Datasets used for speaker recognition systems typically consist of speech recordings from multiple speakers along with their corresponding speaker identities. These datasets are crucial for training and evaluating the performance of speaker recognition systems.

The Dataset consist of voice recordings from 10 different speakers. Each recording is of 5 seconds sampled at 22050 Hz. Each class consist of 4000 samples and split at 60-20-20 for training, validation, and testing, respectively.

The audio recordings are saved at .wav format. Each class recordings are stored in different directories labeled with the name of the respective class.

1.4 SPEAKER RECOGNITION SYSTEM FOR BIO-METRIC APPLICATION

1.4.1 EXISTING SOLUTIONS

One existing solution for a speaker recognition system using GMM is the i-vector approach.

The i-vector approach uses a GMM to model the acoustic properties of speech, where each speaker is represented by a unique i-vector. The i-vector is extracted from the GMM using the Baum-Welch algorithm to estimate the speaker-specific parameters of the model. The i-vector captures the distinctive characteristics of a speaker's voice, such as pitch and speech rate, while also removing speaker-independent information.

The i-vector approach consists of two main stages: training and testing. During the training stage, the GMM is trained on a large dataset of speech recordings from various speakers. The i-vector for each speaker is then extracted from the trained GMM using the Baum-Welch algorithm.

During the testing stage, a speech recording is input into the system, and the system extracts an i-vector from the GMM. The system then compares the i-vector to the i-vectors stored in its database and assigns the input speech recording to the speaker with the closest i-vector.

The i-vector approach has been shown to be effective for speaker recognition tasks, achieving high accuracy rates even in noisy environments. Additionally, it can be used for both speaker verification (determining if a speaker is who they claim to be) and speaker identification (determining the identity of an unknown speaker).

Deep neural networks (DNNs) are another popular approach for speaker recognition systems due to their ability to learn discriminative features from raw audio signals. One existing solution for speaker recognition using DNNs is the x-vector architecture.

The x-vector architecture consists of a convolutional neural network (CNN) followed by a time-delay neural network (TDNN) and a statistical pooling layer. The CNN learns local frequency-domain features from the raw audio waveform, while the TDNN learns temporal patterns across different time frames. The statistical pooling layer aggregates the learned features across time and produces a fixed-length embedding vector, also called the x-vector, which represents the speaker identity.

During training, the x-vector architecture is trained using a softmax loss function to classify the speakers in the training dataset. The learned x-vectors are then used to train a back-end classifier, such as a support vector machine (SVM) or a Gaussian mixture model (GMM), to further improve the speaker recognition performance.

In testing, the x-vector architecture is used to extract the x-vector embeddings from the test audio signals, which are then used as input to the back-end classifier for speaker recognition.

i-vector and x-vector systems assume that the data used for training and testing are generated from a single underlying distribution. However, in real-world scenarios, there can be significant variability in the data due to changes in recording conditions, channel characteristics, and other factors.

1.4.2 PROPOSED SOLUTION

Here, we are proposing a speaker recognition system using Convolutional Neural Network. CNN is trained to recognize unique features in an audio signal that are specific to a particular speaker. The RAW audio data is extracted through the microphone and preprocessed. The features are extracted from the audio signal. The most prominent features are MFCC coefficients. CNN is trained with the MFCC coefficients obtained from the 10 speakers. Model is trained until it reaches the desired accuracy. The system will also have a noise reduction module which remove the noise present in the audio file. Later the classified data is stored in the database.

1.5 SUMMARY

For the chapters to follow, in Chapter 2, the methodology and the drawbacks of the previous approaches which were implemented for the task of speaker recognition are being discussed. In Chapter 3, the detailed description of the proposed solution to overcome the limitations that are being faced by existing solution is being discussed in the detailed manner.

In Chapter 4, the implementation of the proposed solution along with the workflow and the necessary modules that are being implemented for the speaker recognition task are discussed in the detailed manner.

Finally, In the Chapter 5, the result obtained from the experiment is discussed in detailed manner along with the comparison of previously existing solutions result. In Chapter 6, the conclusion of the proposed solution on how it is performing better and being more efficient is being discussed.

LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

Automatic Speaker Recognition Using Gaussian Mixture Speaker Models
Authors: Douglas A. Reynolds

The GMM is a probabilistic model that represents the Gaussian distribution of speech features for a speaker. The speech features can be any acoustic feature that characterizes the speaker's voice, such as mel-frequency cepstral coefficients (MFCCs), spectral features, or prosodic features. The GMM represents the probability distribution of the speech features as a weighted sum of several Gaussian distributions, where each Gaussian represents a distinct feature of the speaker's voice.

During the training phase, a GMM is trained for each speaker using a set of speech recordings. The training data is pre-processed to extract the speech features that are relevant for speaker recognition. For each speaker, a GMM is trained by estimating the parameters of the Gaussian distributions that best fit the speech features of that speaker. The parameters include the mean and covariance of each Gaussian component, as well as the weight of each component in the mixture.

During the testing phase, given an input speech signal, the speech features are extracted and then modelled using the GMMs trained for each speaker. The log-likelihood of the input speech features is calculated for each GMM, and the speaker with the highest likelihood is identified as the speaker of the input speech signal. Experiments conducted in NTIMIT phoneme recognition benchmark using GMM containing 113 speaker is giving overall accuracy of 82.8%.

Advantages of GMM:

1. **Robustness** - GMM-based speaker recognition is robust to variations in speaking style, speaking rate, and channel effects. GMM can model the variability of the speaker's voice due to different factors such as emotion, health condition, and age, making it suitable for real-world applications.

2. **Low complexity:** GMM is computationally efficient and has a low complexity compared to other machine learning algorithms. It can be trained on a small dataset and requires less computational resources during testing.

3. **Flexibility:** GMM can handle multiple speakers in the same recording, making it useful in speaker diarization and segmentation tasks.

4. **Availability of open-source libraries:** There are various open-source libraries available for GMM-based speaker recognition, such as HTK, Kaldi, and Sphinx.

Disadvantages of GMM:

1. **Limited discriminative power:** GMM-based speaker recognition relies on the likelihood ratio between the speaker model and the background model, which may not be sufficient for discriminating between speakers with similar voice characteristics.

2. **Sensitivity to training data:** GMM-based speaker recognition is sensitive to the quality and quantity of training data. It requires a large and diverse dataset to capture the variability in the speaker's voice accurately.

3. **Difficulty in modeling long-term temporal dynamics:** GMM is a static model that cannot capture the temporal dynamics of speech signals over long durations. This limitation can be addressed by incorporating temporal information using dynamic Bayesian networks or deep neural networks.

Speaker Recognition with Deep Recurrent Neural Network
Authors: Alex Graves, Abdel-rahman Mohamed, Geoffrey Hinton.

LSTM is specifically designed to handle long-term dependencies in sequential data. In speech signals, the LSTM network is trained to learn features that capture the speaker's unique voice characteristics such as pitch, tone, and speech cadence.

The input to the LSTM network is a sequence of Mel frequency cepstral coefficients, which are extracted from the speech signal. The LSTM network processes this sequence of MFCCs through multiple layers of LSTM units, which have a memory that enables them to retain information over long periods of time.

The output of the LSTM network is a speaker embedding, which is a vector representation of the speaker's voice characteristics. Experiments conducted in TIMIT phoneme recognition benchmark using Bi-directional LSTM gives 17.4% test error.

Advantages of Deep RNN:

1. Deep RNNs can model long-term temporal dependencies in speech data, making them suitable for speaker recognition applications.

2. Deep RNNs can handle variable-length input sequences, making them suitable for processing speech data, which can have variable lengths.

3. Deep RNNs can be trained end-to-end, allowing them to learn optimal feature representations for the task.

Disadvantages of Deep RNN:

1. Deep RNNs can be computationally expensive to train and require a large amount of training data to achieve optimal performance.

2. Deep RNNs are prone to overfitting, especially when the training data is limited or noisy.

Improved End-to-End Dysarthric Speaker Recognition via Meta-learning Based Model Re-initialization.

Author: Disong Wang, Jianwei Yu, Xixin Wu, Lifa Sun, Xunying Liu, Helen Meng

Hybrid models that combine Gaussian mixture models (GMMs) and deep neural networks (DNNs) have shown promising results for speaker recognition tasks.

GMM-based systems have been widely used for speaker recognition tasks, as they are able to model the complex probability distributions of speech features. However, GMMs have limitations in modeling high-dimensional feature spaces and capturing complex nonlinear relationships between features. On the other hand, DNNs are powerful models that can learn complex feature representations and model complex nonlinear relationships between features.

The GMM component models the probability distribution of the acoustic features, while the DNN component learns discriminative features for speaker recognition. The hybrid model is trained in two stages. In the first stage, a GMM is trained using a large dataset of speech features. In the second stage, a DNN is trained on top of the GMM output to learn a discriminative feature representation for speaker recognition. Experimental results on UASpeech dataset show that the best model with proposed methods achieves 54.2% and 7.6% relative word error rate reduction compared with the base model without finetuning and the model directly fine-tuned from the base model, respectively.

Advantages:

1. The hybrid model combines the strengths of GMMs and DNNs, resulting in better performance than either model alone.

2. The GMM component provides a strong prior on the distribution of speech features, which can improve the robustness of the system to variations in speech data.

3. The DNN component can learn a more discriminative feature representation for speaker recognition, improving the overall performance of the system.

Disadvantages:

1. The hybrid model requires more training data and computational resources than a standalone GMM or DNN model.

2. The hybrid model is more complex than a standalone GMM or DNN model, making it more difficult to implement and maintain.

Deep Learning Backend for Single and Multisession i-Vector Speaker Recognition
Author: Omid Ghahabi and Javier Hernando

For multi-session i-vector speaker recognition, deep learning can be used to model the variability in speaker characteristics over time. This can be achieved using techniques such as deep neural network (DNN) adaptation, where a DNN is trained on a large dataset of speech signals to model speaker variability. The DNN is then trained with specific speaker using a smaller amount of speech data from that speaker. The adapted DNN can then be used to extract speaker-specific features from new speech signals, which can be combined with the i-vector for classification. Experiments conducted on NIST 2014 i-vector challenge gives a improved accuracy of 89.4%.

Advantages:

1. Deep learning backends have been shown to outperform traditional GMM-UBM backends in terms of speaker recognition accuracy, especially in challenging scenarios where there is a large mismatch between the training and testing data.

2. With a deep learning backend, the entire system can be trained end-to-end, including the i-vector extractor and the speaker classifier. This allows for better optimization of the entire system and can lead to improved performance.

3. Deep learning backends are highly scalable, allowing for the efficient processing of large datasets.

Disadvantages:

1. Deep learning backends typically require large amounts of labeled training data, which can be a challenge in some applications, especially for languages with limited resources.

2. Deep learning backends can be vulnerable to adversarial attacks and can exhibit unpredictable behavior in scenarios outside of the training data distribution. This can be a challenge in security-sensitive applications.

PROJECT DESCRIPTION

CHAPTER 3

PROJECT DISCRIPTION

3.1 PROPOSED SYSTEM

We are proposing the 3 layered architecture system for speaker recognition-based Bio-Metric system. Where the edge layer or Layer 1 consist of audio recordings, noise reduction module, feature extraction module and CNN model for speaker recognition. The output from the edge layer which is class identified by speaker recognition module is given as the input to the server or database layer which the corresponding class is updated into the database with time and data. The layer 3 is a web layer where we can be able to visualize the database. The webpage can facilitate the input of date, which returns the person who are all present in that date.

3.2 OVERALL ARCHITECTURE

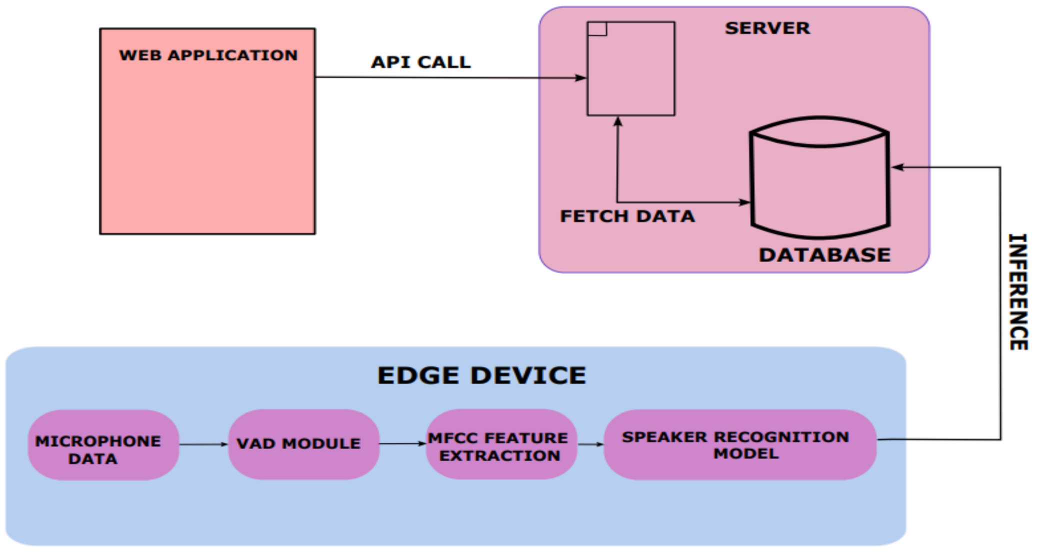


Figure 3.1: Represents the overall architecture for speaker recognition system.

Figure 3.1 represents the Three-Layered Architecture that has been implemented for the speaker recognition system. Edge Module represents the sequence of events occurring at layer 1 which involves data collection, noise reduction, feature extraction followed by classification the output class later send to database.

3.3 LAYER 1 (EDGE LAYER)

The edge layer in a system responsible for audio recording, noise reduction, feature extraction, and classification using CNN is an essential component of the overall architecture. This layer is designed to handle the initial processing of audio signals captured by microphones and then perform analysis on the signal to identify specific features that can be used for further processing. The recorded audio signal is then passed to the next stage in the pipeline, which is noise reduction.

Noise reduction is an important step in the pipeline because it helps to remove any unwanted background noise from the audio signal. This can be done by spectral gating approach. The goal of this stage is to produce a clean audio signal that can be used for further processing.

After noise reduction, the next step is feature extraction. This is where the audio signal is analysed to identify specific features that can be used for classification. In speech recognition, for example, features such as Mel frequency cepstral coefficients (MFCCs) are commonly used. These features are extracted from short segments of the audio signal and are used to capture information about the frequency content of the signal.

Once the features have been extracted, the next step is classification using CNN. Convolutional neural networks (CNNs) are commonly used for audio classification tasks because they can learn complex features from raw audio signals. The CNN takes the extracted features as input and uses a series of convolutional layers to learn representations of the signal. These representations are then passed to fully connected layers for final decision.

The output from the edge layer is then passed to the server layer for further processing. This includes aggregation of results from multiple edge devices. The server layer can also be used to store and analyse the data collected from the edge devices.

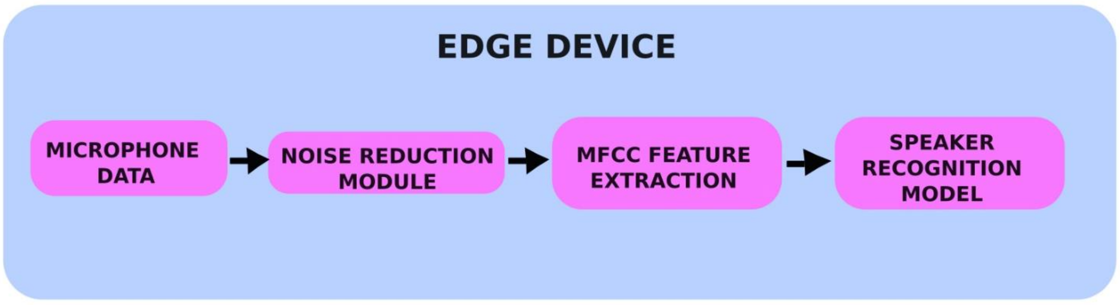


Figure 3.2: Represents the sequence of events happening at edge layer.

Figure 3.2 Represents the sequence of events happening at edge layer that involves Audio recording through microphone and performing noise reduction which will be followed by MFCC feature extraction later the extracted MFCC features will be later given as input to Speaker recognition module.

3.4 LAYER 2 (SERVER LAYER)

The server layer is the second component of the overall system architecture that acts as a mediator between the edge layer and the database. In the context of the given scenario, the server layer receives the classified speaker class from the edge layer and updates the class name, date, and time in the MySQL database.

Once the edge layer has completed the task of classifying the speaker, it sends the output to the server layer. The server layer then performs the necessary processing to store this information in the database. This typically involves updating an existing record or creating a new one if the speaker is not already existing.

To interact with the database, the server layer uses a database management system (DBMS) such as MySQL. The DBMS provides a way for the server layer to store and retrieve data in an efficient and reliable manner.

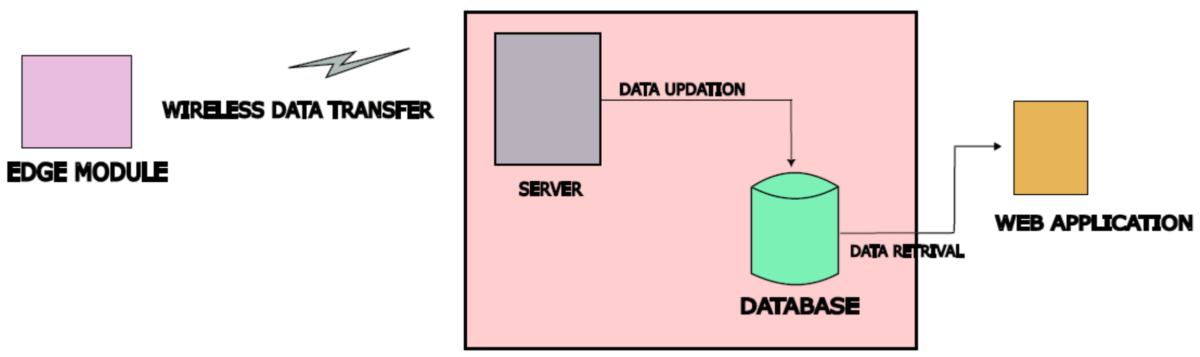


Figure 3.3: Represents the sequence of events happening at server layer.

Figure 3.3 Represents the Data Updation and API calls occurring in server later. The user can send the date as the input from the webpage. The Data that the user sent will be used to query the database. The obtained result will be displayed in the webpage. The result obtained in the edge module will be sent to the server wirelessly and get updated in the database.

3.5 LAYER 3 (WEB LAYER)

The web layer is a third component of overall system architecture. It is responsible for receiving requests from users and responding with the appropriate data. In the case of a web application that needs to return a list of people present on a particular date, the web layer can receive the date as input and query the database to retrieve the necessary information.

Once the date is received by the web layer, it can query the database to retrieve a list of people who are present on that date.

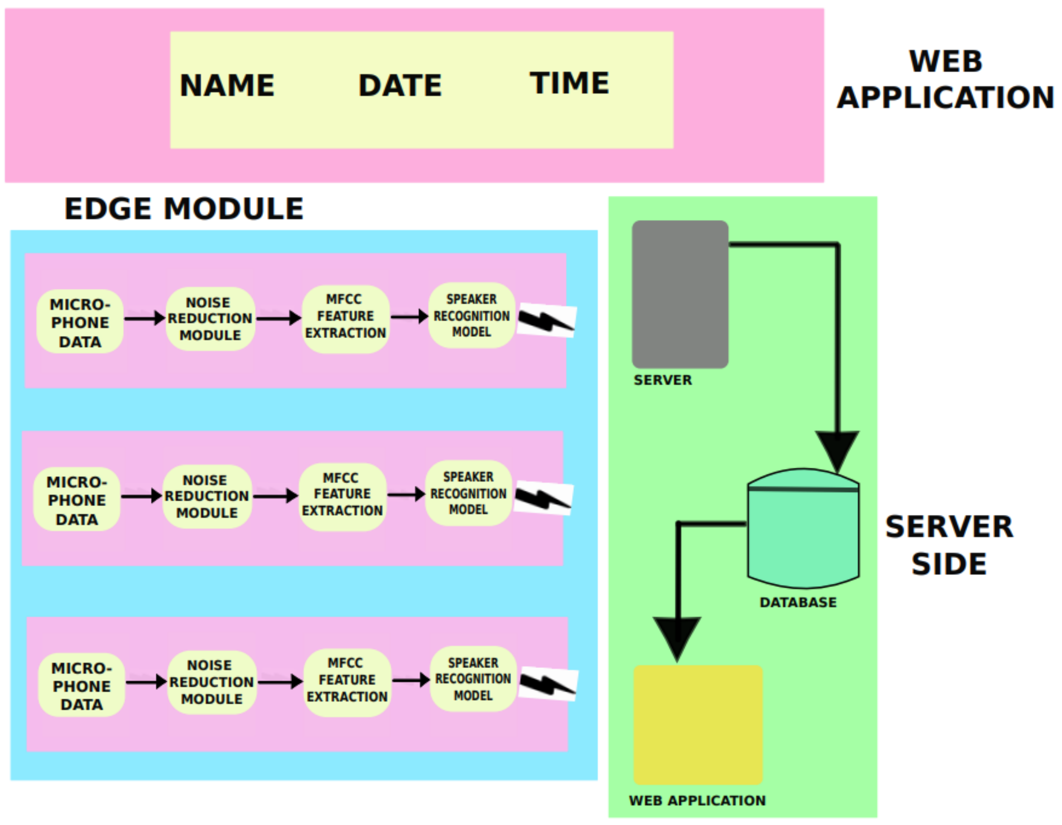


Fig 3.4 Represents the entire architecture of speaker recognition system.

Figure 3.4 Represents the entire architecture implemented for the speaker recognition system. The are several edge modules that can be implemented, and all performs the classification task. Here we are restricting the computations in the edge level. The result obtained from the edge layer is sent to the centralized server. The server is hosting the MySQL database which maintains the record of individuals present in any day. The admin can be able to look upon the records of the database. He can be able to give date as the input and look up who are all present in the date. The database will be queried using the date and results are displayed in the web page.

IMPLEMENTATION

CHAPTER 4

IMPLEMENTATIONS

4.1 NOISE REDUCTION USING SPECTRAL GATING

Noise Reduction can be done by using spectral gating technique. It involves analyzing the frequency content of an audio signal and selectively attenuating or eliminating portions of the signal that contain unwanted noise or other unwanted sounds. The basic idea behind spectral gating is to set a threshold for each frequency band in the signal. If the energy in each frequency band falls below the threshold, that frequency band is "gated" or attenuated, effectively removing any noise or unwanted sounds that may be present in that band.

Spectral Gating based noise reduction works by works by computing a spectrogram of a signal (and optionally a noise signal) and estimating a noise threshold (or gate) for each frequency band of that signal/noise. That threshold is used to compute a mask, which gates noise below the frequency-varying threshold.

The two versions for spectral gating algorithm are:

1. **Stationary Noise Reduction**: Keeps the estimated noise threshold at the same level across the whole signal.

2. **Non-stationary Noise Reduction**: Continuously updates the estimated noise threshold over time.

Here, we have implemented a Non-stationary Noise Reduction method in which the noise threshold continuously updates over time. The values present in the filters get updated in time which facilities the non-continuous noise removal and results in clean noise free audio.

4.1.1 STEPS INVOLVED IN NOISE REDUCTION

1. An FFT is calculated over the noise audio clip.
2. Statistics are calculated over FFT of the noise (in frequency).
3. A threshold is calculated based upon the statistics of the noise.
4. An FFT is calculated over the signal.
5. A mask is determined by comparing the signal FFT to the threshold.
6. The mask is smoothed with a filter over frequency and time.
7. The mask is applied to the FFT of the signal and inverted.

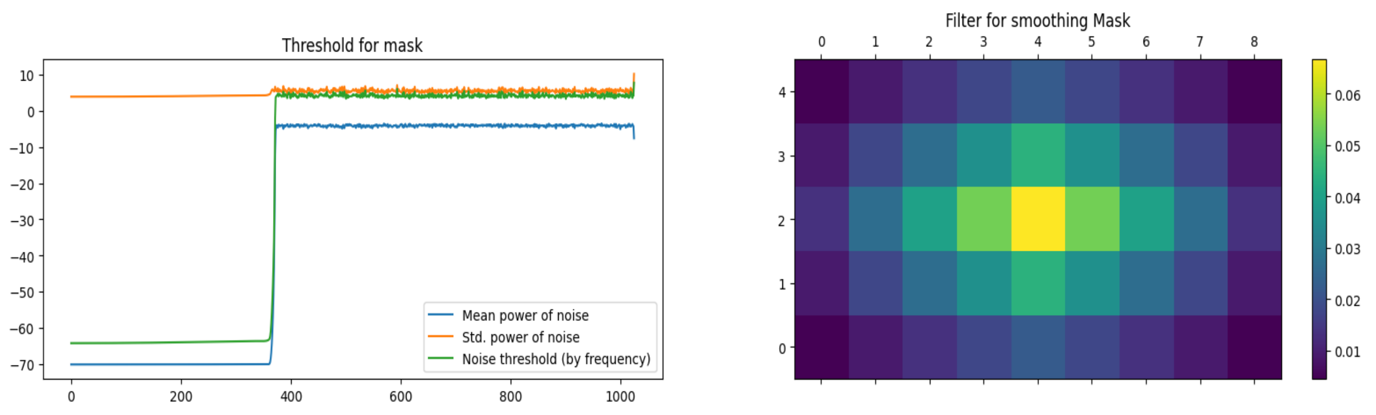


Fig 4.1: Represents the Threshold that applied for Mask.

Figure 4.1 Represents the statistics that has been computed for calculating the threshold value for noise reduction. Once the statical values like mean and standard deviations are calculated the filter values got updated and FFT convolutions is performed. The same repeated for every frame of the audio signal since we are performing non-stationary noise removal.

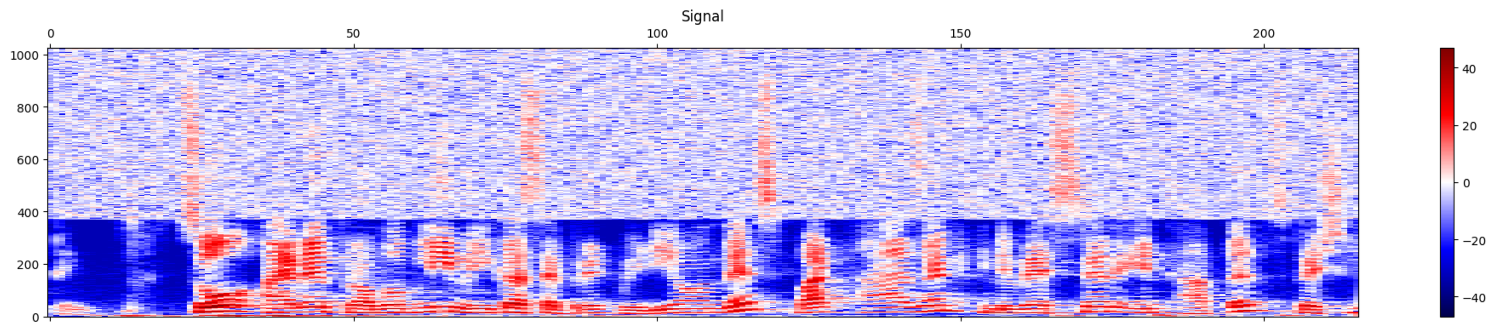


Fig 4.2: Represents the spectrogram of signal with Noise.

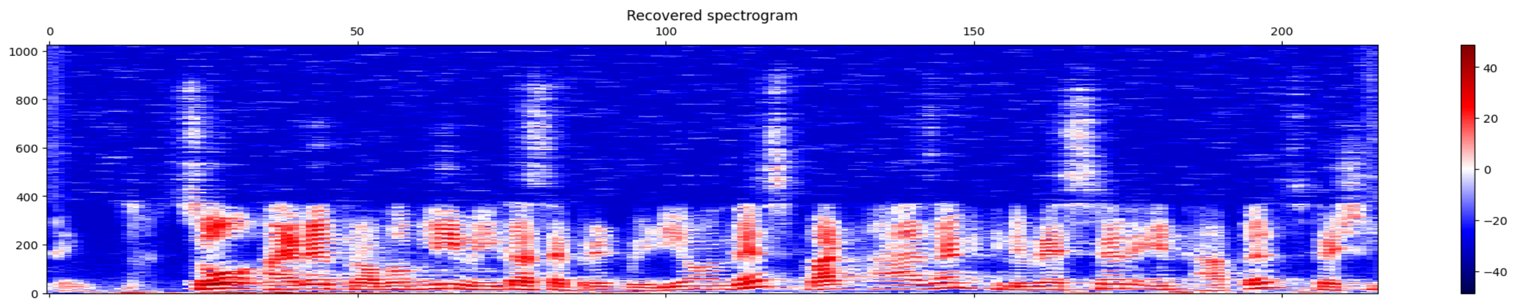


Fig 4.3: Represents the spectrogram of reconstructed signal.

Figure 4.2 and Figure 4.3 represents the spectrogram of the signal with noise and reconstructed signal, respectively. The noised signal has more distortion in its frequency component and the reconstructed signal is less distorted.

4.2 MFCC FEATURE EXTRACTION

Mel-frequency cepstral coefficients (MFCCs) are widely used for speech signal processing and analysis. MFCCs are a feature extraction technique that involves converting speech signals into a sequence of cepstral coefficients, which are then used for various applications such as speaker recognition.

MFCCs are based on the human auditory system's response to sound, which is modeled using the Mel frequency scale. This scale is designed to reflect the non-linear nature of human perception of sound frequency, where the perceived difference between two frequencies is smaller at higher frequencies than at lower frequencies.

4.2.1 STEPS INVOLVED IN MFCC FEATURE EXTRACTION:

STEP 1 – Split the signal into shorter frames.

We need to split the signal into short-time frames. The rationale behind this step is that frequencies in a signal change over time, so in most cases it does not make sense to do the Fourier transform across the entire signal in that we would lose the frequency contours of the signal over time. Frame the signal into 25 ms frames. This means the frame length of our 22050 Hz signal is.

$$\mathbf{0.025*22050 = 551\ samples}$$

with a sample hop length of 256 samples.

STEP 2 - **Windowing**

Windowing is essentially applied to notably counteract the assumption made by the Fast Fourier Transform that the data is infinite and to reduce spectral leakage.

STEP 3 - **Calculation of Discrete Fourier Transform**

Apply 2048-point FFT which is Short Time Fourier Transformation on every frame to obtain the frequency spectrum.

STEP 4 – Applying Filter Banks

Apply Mel spaced filter banks. Filter Banks contains 13 vector of length 2048. By multiplying the filter bank with the frequency spectrum, we will obtain the spectral bank energy present in the audio.

Our filter bank comes in the form of 13 vectors of length 2048. Each vector is mostly zeros but is non-zero for a certain section of the spectrum. To calculate filter bank energies, we multiply each filter bank with the power spectrum, then add up the coefficients. Once this is performed, we are left with 40 numbers that give us an indication of how much energy was in each filter bank.

STEP 5 - Apply log to the obtained spectrogram values to get log filter bank energies.

STEP 6 – Filter bank coefficients are highly correlated, Apply DCT to obtain uncorrelated Mel spectrum.

At the end we obtained Mel coefficient vector of size 13*216 is obtained for every 5 seconds of audio samples.

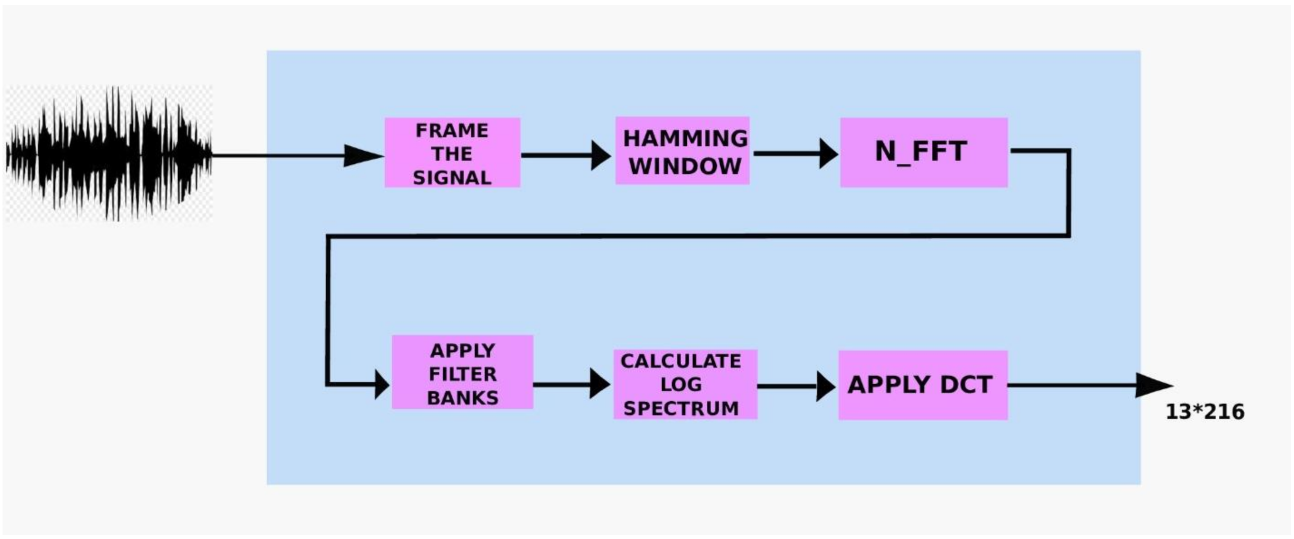


Figure 4.4: Represents the sequence of steps involved in MFCC Feature Extraction.

Figure 4.4 Represents the sequence of events involved in MFCC feature extraction from framing the signals followed by windowing to smoothen the frequency component then applying N point FFT to calculate the filter bank energies, the filter bank energies represents the actual information present in the spectrum higher the filter bank energies higher the information present, then applying log to the compute log spectrum to obtain log power spectrum, finally followed by applying DCT to obtain the uncorrelated features.

4.2.2 CALCULATIONS

DURATION OF AUDIO SAMPLES: **5 SECS**

SAMPLING RATE: **22050**

NO OF MFCC COEFFICIENTS: **13**

HOP LENGTH: **512**

N_FFT: **2048**

TOTAL NO OF SAMPLES IN A SINGLE AUDIO FILE: **110250 SAMPLES**

TOTAL NO OF FRAME: **110250/512 = 216 FRAMES**

For each frame **13 MFCC** coefficients are calculated, thus finally for a single audio file we are obtaining **13*216 MFCC** coefficients.

The filter banks using in calculating the Mel coefficients contains 13 vectors of size 2048. Since, we are calculating the 13 n_mfcc component 13 vectors have been used. Each vector is of length 2048. The vector mostly contains zeros but is non-zero for a certain section of the spectrum. To calculate filter bank energies, we multiply each filter bank with the power spectrum, then add up the coefficients. Once this is performed, we are left with 13 numbers that give us an indication of how much energy was in each filter bank.

4.2.3 MFCC HEATMAP OF VARIOUS CLASSES.

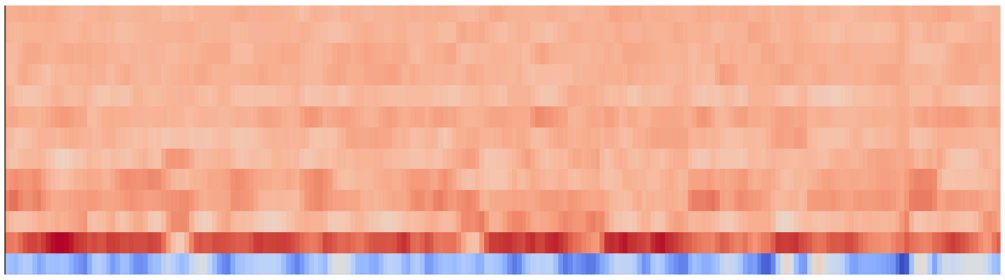


Fig 4.5 (a): Represents the heatmap of mfcc coefficients of class Akshat.

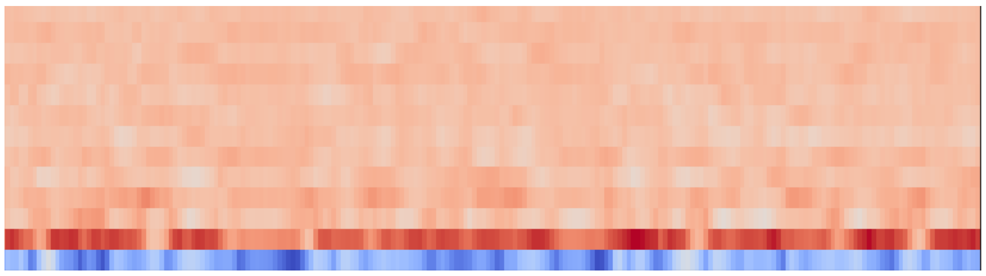


Fig 4.5 (b): Represents the heatmap of mfcc coefficients of class GeekyRanjit.

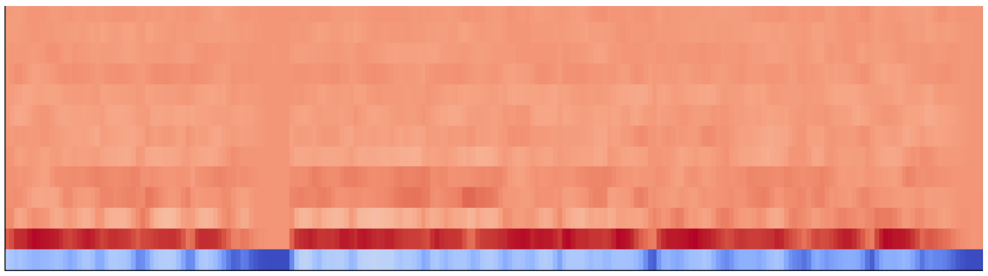


Fig 4.5 (c): Represents the heatmap of mfcc coefficients of class Jeyaprathap.

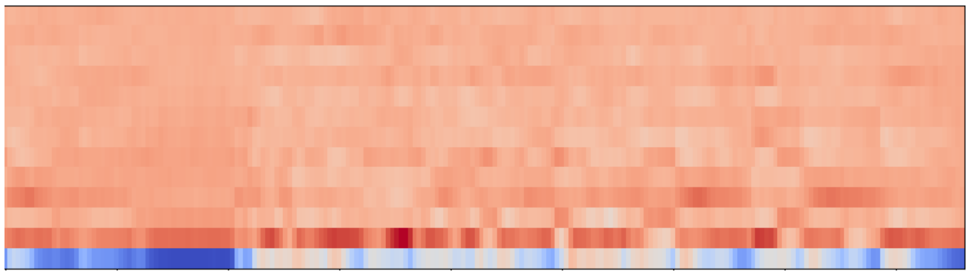


Fig 4.5 (d): Represents the heatmap of mfcc coefficients of class mkbhd

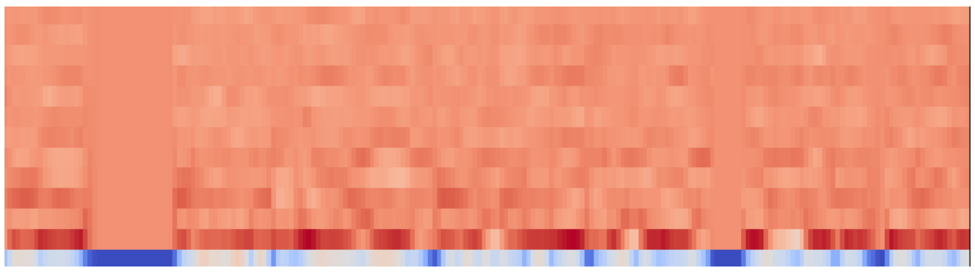


Fig 4.5 (e): Represents the heatmap of mfcc coefficients of class Mosh.

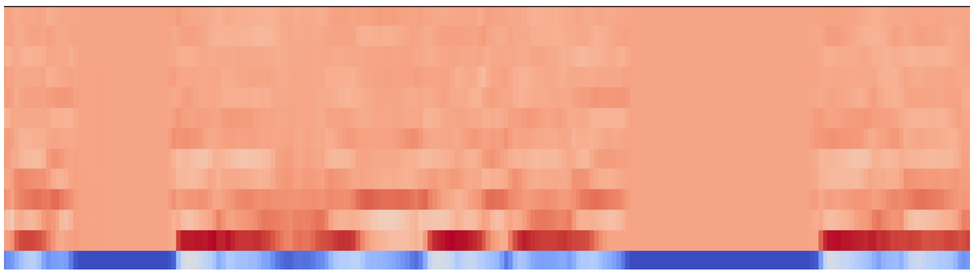


Fig 4.5 (f): Represents the heatmap of mfcc coefficients of class Paramesh.

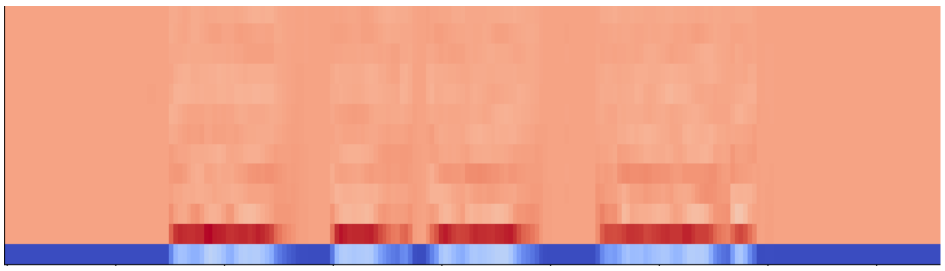


Fig 4.5 (g): Represents the heatmap of mfcc coefficients of class Rahul.

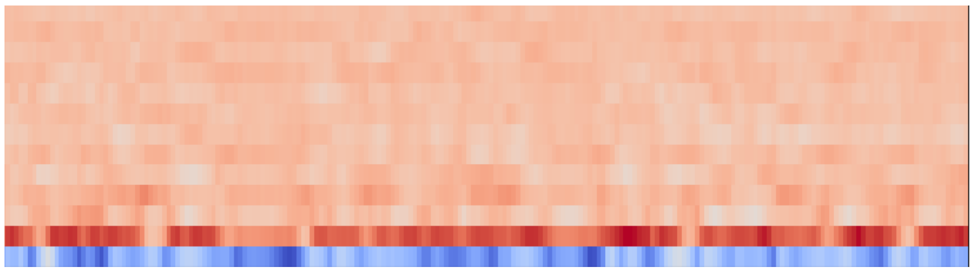


Fig 4.5 (h): Represents the heatmap of mfcc coefficients of class C4ETech.

Figure 4.5 (a) to Figure 4.5 (h) represents the heatmap of MFCC coefficients. The pink color regions represent the spectral energy concentrated in low frequency regions which is having the positives values and the blue color region represents the spectral energy concentrated in high frequency region that are having negative values. There is certain region in which the heatmap is not showing significant changes in frequency that represent the lack of energy present in the sequence of signal.

4.3 IMPLEMENTATION OF CONVNET FOR SPEAKER RECOGNITION TASK

The Convolution neural network implemented for speaker recognition task of speaker recognition have following layers. The detailed configuration of the layers will be explained later in the following sections.

1. **Input layer:** This layer takes the 13 x 216 input data and feeds it into the network.

2. **Convolutional layers:** These layers perform convolutions on the input data, using a set of filters that learn to recognize specific patterns in the data. Each filter produces a feature map that represents the presence of the pattern at different locations in the input data.

3. **Pooling layers:** These layers down sample the feature maps produced by the convolutional layers, reducing their size while retaining the most important information.

4. **Fully connected layers:** These layers take the output of the previous layers and use it to classify the input data into different categories.

5. **Output layer:** This layer produces the final output of the network, which could be a probability distribution over the different categories, or a set of continuous values.

4.3.1 DETAILED CNN CONFIGURATION:

CONV2D LAYER

The CNN architecture has an input shape of (216, 13, 1), which means it takes input in the form of a tensors with 216 rows and 13 columns, with a channel. The architecture consists of a sequence of layers stacked together, and each layer performs a specific operation on the input.

The first layer of the network is a 2D convolutional layer (Conv2D), which uses 64 filters of size 3x3 to extract features from the input image. The output of this layer has a shape of (None, 216, 13, 64), where "None" represents the batch size.

The calculation for the output shape is as follows:

output_height = input_height - kernel_height + 1
output_width = input_width - kernel_width + 1
output_depth = number_of_filters

output_shape = (None, output_height, output_width, output_depth)

In this case, the output height and width are calculated as follows:

output_height = 216 - 3 + 1 = 214
output_width = 13 - 3 + 1 = 11

Therefore, the output shape of the Conv2D layer is (None, 214, 11, 64).

The number of trainable parameters in this layer is calculated as follows:

number_of_params = (kernel_height * kernel_width * input_depth + 1) * number_of_filters

In this case, the number of trainable parameters is:

number_of_params = (3 * 3 * 1 + 1) * 64 = 640

MAXPOOLING2D LAYER

The next layer is a max pooling layer (MaxPooling2D) that reduces the spatial dimensions of the output from the previous layer by a factor of 2, resulting in an output shape of (None, 108, 6, 64).

The calculation for the output shape is as follows:

The next layer is a max pooling layer (MaxPooling2D) that reduces the spatial dimensions of the output from the previous layer by a factor of 2, resulting in an output shape of (None, 108, 6, 64).

The calculation for the output shape is as follows:

output_height = input_height / pool size
output_width = input_width / pool size
output_shape = (None, output_height, output_width, input depth)

In this case, the output shape is:

output_height = 216 / 2 = 108
output_width = 13 / 2 = 6

Therefore, the output shape of the MaxPooling2D layer is (None, 108, 6, 64).

BATCH NORMALIZATION LAYER

The next layer is a batch normalization layer (Batch Normalization), which normalizes the output of the previous layer to improve the performance of the network. This layer has 256 trainable parameters, which are used to scale and shift the normalized output.

The normalization equation for a single channel is:

x_norm = (x - mean) / sqrt(variance + epsilon)

where **x** is the input, **mean** is the mean of the input, **variance** is the variance of the input, and **epsilon** is a small constant added to the variance to avoid division by zero.

The scaling equation is:

y = gamma * x_norm + beta

where **y** is the normalized and scaled output, **gamma** is a learnable scaling factor, and **beta** is a learnable shift factor.

CONV2D LAYER 2, 3

The next two layers are like the first three layers, with the only difference being the number of filters used in the convolutional layers. The second Conv2D layer has 32 filters of size 3x3, and the third Conv2D layer has 16 filters of size 3x3.

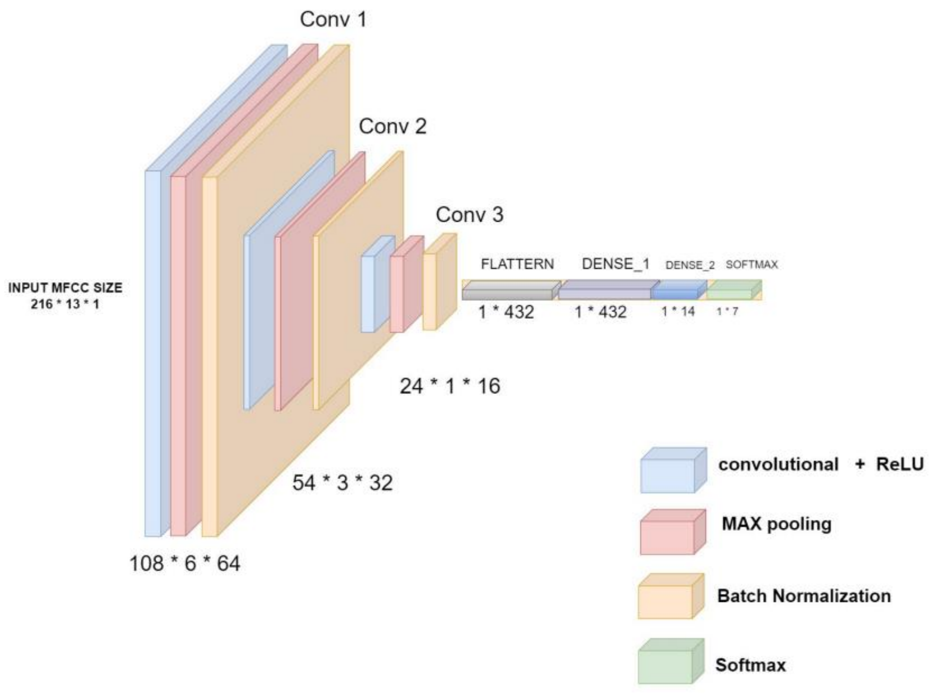


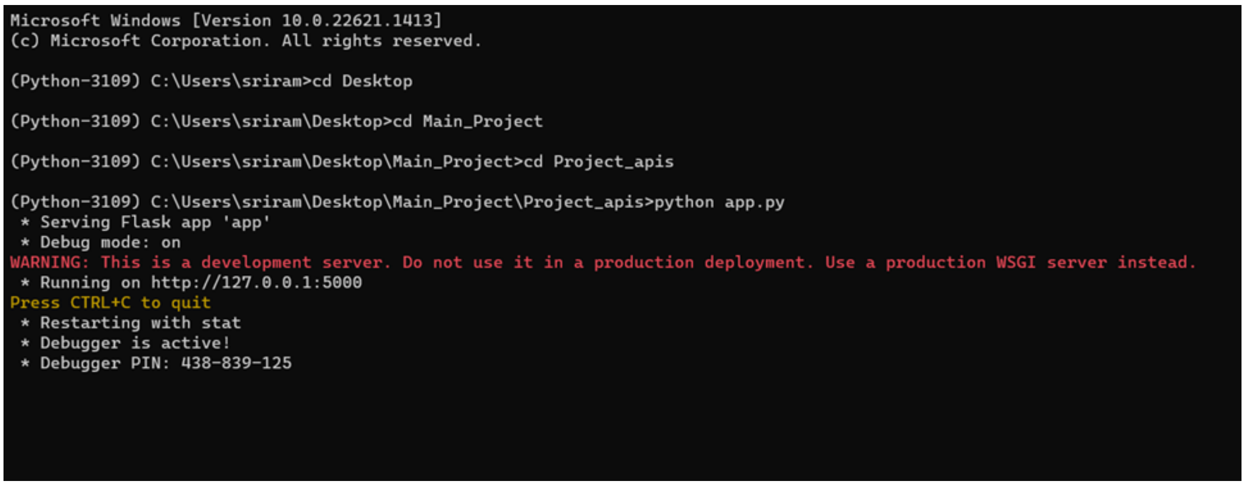
Figure 4.7: CNN Architecture implemented for speaker recognition system.

Figure 4.7 Represents the CNN architecture that has been implemented for speaker recognition system. The model takes MFCC features as the input, and it outputs the probability of 10 classes. There Three convolution layer in total each layer is followed by max pooling and batch normalization.

4.4 IMPLEMENTATION OF CLIENT SERVER MODEL

The client-server-based architecture with a Raspberry Pi and Windows computer, where the Raspberry Pi sends classified audio class using deep learning to Windows, which acts as a server wirelessly, can be extended to update the corresponding class in a MySQL database.

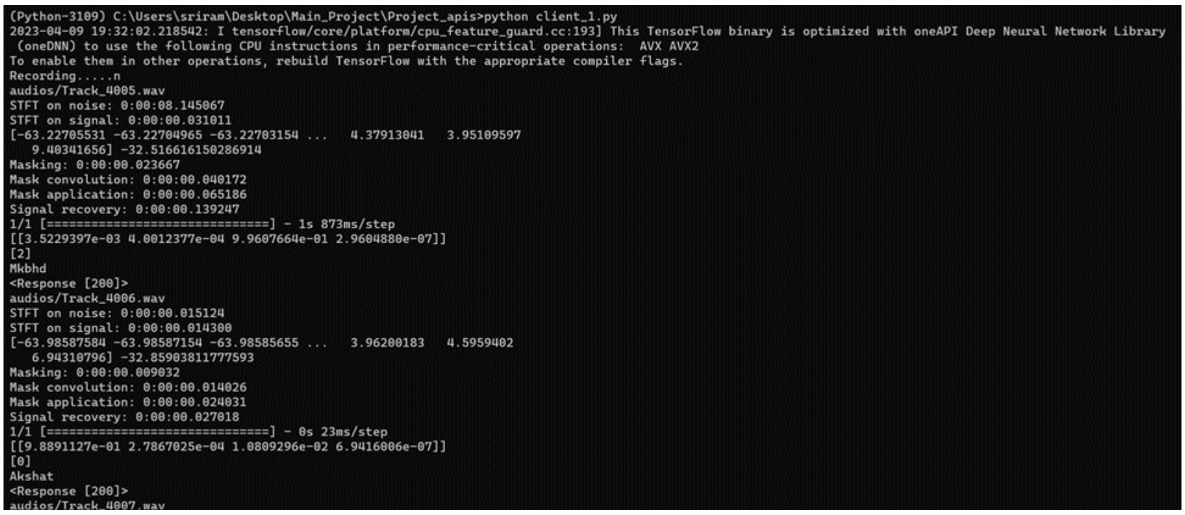
To set up the client-server architecture, we first need to configure the Raspberry Pi and the Windows computer for wireless communication. We can use a Wi-Fi network to establish a connection between the two devices. The Raspberry Pi can capture audio data using a microphone and preprocess it is using various audio signal processing techniques like normalization, filtering, and feature extraction. The preprocessed audio data can be classified using a deep learning model trained on a dataset of audio files that are labeled with their corresponding categories. The classified audio data can be sent to the windows computer using a network protocol like TCP/IP. The windows computer can receive the data, parse it, and update the corresponding class in a MySQL database using a Python script.



```
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

(Python-3109) C:\Users\sriram>cd Desktop

(Python-3109) C:\Users\sriram\Desktop>cd Main_Project

(Python-3109) C:\Users\sriram\Desktop\Main_Project>cd Project_apis

(Python-3109) C:\Users\sriram\Desktop\Main_Project\Project_apis>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 438-839-125
```

Fig 4.8 Flask Application running on Localhost.

Figure 4.8 represents the Flask application running on the local host IP 127.0.0.1:5000. The webpage can be accessed from the IP which the application is running on. And we can send the classified data from the edge module to server by including update_data route with the IP the application is running on.



```
(Python-3109) C:\Users\sriram\Desktop\Main_Project\Project_apis>python client_1.py
2023-04-09 19:32:02.218542: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
 (oneDNN) to use the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Recording.....n
audios/Track_4005.wav
STFT on noise: 0:00:08.145067
STFT on signal: 0:00:00.031011
[-63.22705531 -63.22704965 -63.22703154 ...   4.37913041   3.95109597
   9.40341656] -32.516616150286914
Masking: 0:00:00.023667
Mask convolution: 0:00:00.040172
Mask application: 0:00:00.065186
Signal recovery: 0:00:00.139247
1/1 [==============================] - 1s 873ms/step
[[3.5229397e-03 4.0012377e-04 9.9607664e-01 2.9604880e-07]]
[2]
Mkbhd
<Response [200]>
audios/Track_4006.wav
STFT on noise: 0:00:00.015124
STFT on signal: 0:00:00.014300
[-63.98587584 -63.98587154 -63.98585655 ...   3.96200183   4.5959402
   6.94310796] -32.85903811777593
Masking: 0:00:00.009032
Mask convolution: 0:00:00.014026
Mask application: 0:00:00.024031
Signal recovery: 0:00:00.027018
1/1 [==============================] - 0s 23ms/step
[[9.8891127e-01 2.7867025e-04 1.0809296e-02 6.9416006e-07]]
[0]
Akshat
<Response [200]>
audios/Track_4007.wav
```

Fig 4.9: classified data from client

The Figure 4.9 represents the sequence of event happing at edge level in a terminal view. That involves audio recording, noise reduction, classification and sending the classified data to the server. Once the class in classified by the model it is then sent to the server and get updated in the database. The response 200 HTTP code represents that the data is transferred successfully and it is received at the server end. If the data is not send properly then we will get response 500 HTTP code represents the Internal Server Error server error response code indicates that the server encountered an unexpected condition that prevented it from fulfilling the request.

4.5 IMPLEMENTATION OF RELATIONAL DATABASE

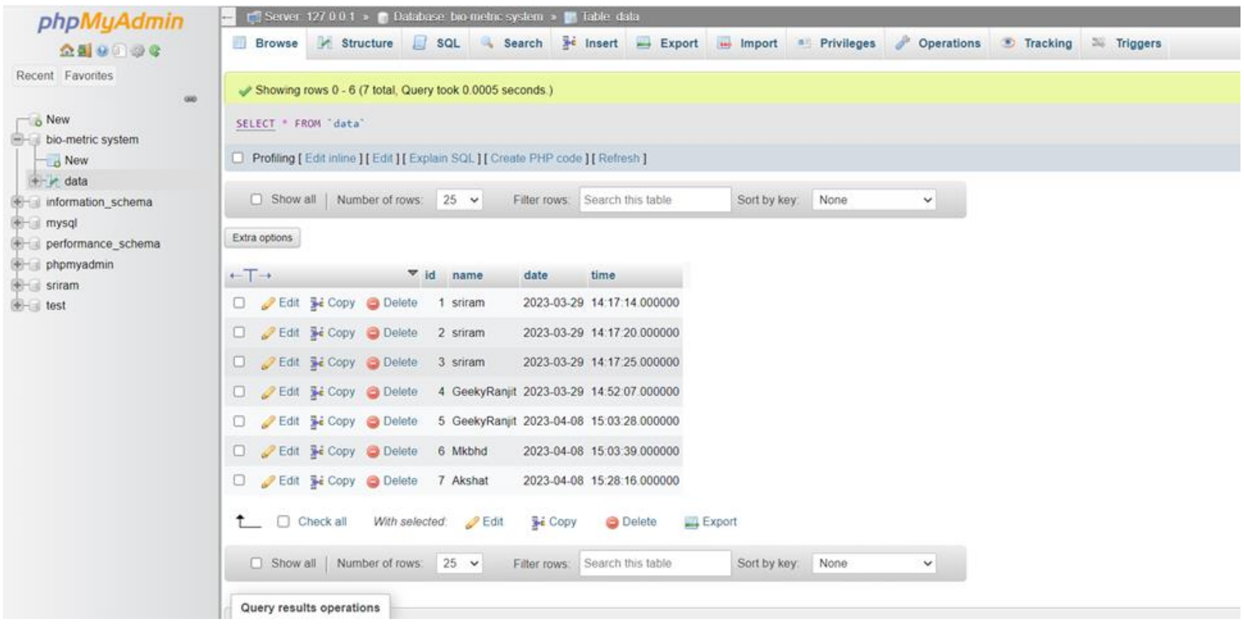
A relational database can be an effective tool for managing data in a speaker recognition-based biometric system. The database can contain attributes like name, ID, date, and time, which can be used to track and manage speaker recognition data.

To begin with, the database should have a table to store speaker recognition data. This table can have several attributes, including name, ID, date, time, and biometric data. The name attribute can store the name of the speaker, while the ID attribute can store a unique identifier for the speaker. The date and time attributes can store the date and time when the biometric data was captured, while the biometric data attribute can store the biometric data used for speaker recognition, such as voiceprints or speech patterns.

The database can be designed using a relational database management system like MySQL, which is an open-source relational database management system. The system can be designed using a graphical interface, which provides a user-friendly interface for querying databases.

The database can be used to store and manage speaker recognition data from various microphones. The data can be captured at different times and dates and can be used to identify speakers and authenticate their identity.

The database can also be used to store and manage historical data, which can be used for analysis and decision-making. For example, the database can be used to analyze patterns in speaker recognition data and identify trends or anomalies. This can help in identifying potential security threats or identifying speakers with similar biometric data.



Showing rows 0 - 6 (7 total, Query took 0.0005 seconds.)

```
SELECT * FROM `data`
```

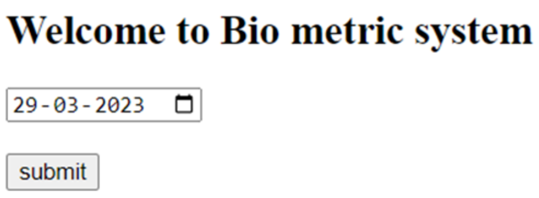
id	name	date	time
1	sriram	2023-03-29	14:17:14.000000
2	sriram	2023-03-29	14:17:20.000000
3	sriram	2023-03-29	14:17:25.000000
4	GeekyRanjit	2023-03-29	14:52:07.000000
5	GeekyRanjit	2023-04-08	15:03:28.000000
6	Mkbhd	2023-04-08	15:03:39.000000
7	Akshat	2023-04-08	15:28:16.000000

Fig 4.9 Database for storing classified speakers.

Figure 4.9 Represents the database schema that has been implemented for the storing the name of the persons. The table has three main attributes name, date, and time. The name column represents the name of the person and date and time represent the date and time in which the person was present. Which updating the database with the person's name first we will check if any entries present in the table with same name and time if there is no such entries exist we will create a new entry.

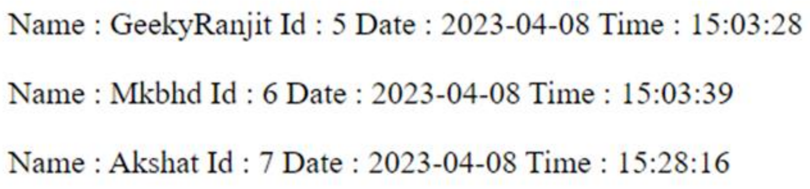
4.6 IMPLEMENTATION OF WEB APPLICATION

The web application facilitates the input of date. The user can give the date as an input and the result the displayed in the web page. Once the date is submitted in the form through the POST request the database is queried based on the date. The result is fetched from the table, and it will be displayed in the webpage. The result includes Name, Date and Time.


Welcome to Bio metric system

29-03-2023

Fig 4.10: Homepage of web application


Name : GeekyRanjit Id : 5 Date : 2023-04-08 Time : 15:03:28

Name : Mkbhd Id : 6 Date : 2023-04-08 Time : 15:03:39

Name : Akshat Id : 7 Date : 2023-04-08 Time : 15:28:16

Fig 4.11: Results obtained from the database.

Figure 4.10 and Figure 4.11 represent the actions that can be performed in a web page. Figure 4.10 represents the homepage of the application which facilitates the input of date through html form. Date can be given as input. The Figure 4.11 represents the result obtained from the query. The database is query upon date and results includes Name, Date and Time.

RESULTS

CHAPTER 5

RESULTS

Our proposed CNN architecture for speaker recognition-based Bio-Metric system has achieved best result.

speaker recognition system using CNN that achieves an accuracy of 98.2% on testing data is a highly accurate system. The system has demonstrated a high level of effectiveness in correctly recognizing and identifying speakers, which can be useful in a variety of applications such as security, authentication, and speech processing.

The system is testing on 800 samples of 10 classes containing 80 samples in each class. The system shows improved performance than traditional speaker recognition algorithm such as GMM-UBM which is having the accuracy of only 89.02%.

The testing data used to evaluate the system represents the possible test cases and conditions that the system may encounter in real-world applications. And the proposed system is robust to noise. The system accuracy is accessed under variety of conditions such as with and without noise to ensure its reliability and effectiveness in practical applications.

CONCLUSION

CHAPTER 6

CONCLUSION

The implementation of a speaker recognition biometric system using Convolutional Neural Networks (CNNs) shows promising results. The CNN architecture has shown great success in computer vision applications and its adaptation to speaker recognition has yielded excellent results. The system's ability to accurately identify individuals based on their unique vocal features can have numerous practical applications in security, law enforcement, and communication industries.

Moreover, the development of such systems can be a significant step towards a safer and more secure future. The utilization of deep learning techniques like CNNs can significantly improve the accuracy and efficiency of biometric systems, leading to increased trust in their use. However, further research and development in this area are still needed to enhance the performance of speaker recognition systems and their robustness against adversarial attacks.

Overall, the speaker recognition biometric system using CNN is a promising technology that can potentially revolutionize the security and communication industries. Its successful implementation can provide a reliable and secure means of identification, and thus, improve the safety and security of individuals and organizations alike.

REFERENCES

REFERENCE

1. N. N. An, N. Q. Thanh and Y. Liu, "Deep CNNs With Self-Attention for Speaker Identification," in IEEE Access, vol. 7, pp. 85327-85337, 2019, doi: 10.1109/ACCESS.2019.2917470.

2. I. Shahin, A. B. Nassif and S. Hamsa, "Emotion Recognition Using Hybrid Gaussian Mixture Model and Deep Neural Network," in IEEE Access, vol. 7, pp. 26777-26787, 2019, doi: 10.1109/ACCESS.2019.2901352.

3. Hourri, S., Kharroubi, J. A deep learning approach for speaker recognition. *Int J Speech Technol* **23**, 123–131 (2020).

4. M. M. Kabir, M. F. Mridha, J. Shin, I. Jahan and A. Q. Ohi, "A Survey of Speaker Recognition: Fundamental Theories, Recognition Methods and Opportunities," in IEEE Access, vol. 9, pp. 79236-79263, 2021, doi: 10.1109/ACCESS.2021.3084299

5. Hyeong-Seok Choi and Janghyun Kim and Jaesung Huh and Adrian Kim and Jung-Woo Ha and Kyogu Lee," Phase-Aware Speech Enhancement with Deep Complex U-Net": International Conference on Learning Representations-2019

6. A. Graves, A. -r. Mohamed and G. Hinton, "Speech recognition with deep recurrent neural networks," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 2013, pp. 6645-6649, doi: 10.1109/ICASSP.2013.6638947.

7. D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," in *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 1, pp. 72-83, Jan. 1995, doi: 10.1109/89.365379.

8. Tara N. Sainath, Brian Kingsbury, George Saon, Hagen Soltau, Abdel-rahman Mohamed, George Dahl, Bhuvana Ramabhadran, Deep Convolutional Neural Networks for Large-scale Speech Tasks.

9. Qin, C.X., Qu, D. and Zhang, L.H., 2018. Towards end-to-end speech recognition with transfer learning. *EURASIP Journal on Audio, Speech, and Music Processing*, *2018*(1), pp.1-9.

10. D. Wang, J. Yu, X. Wu, L. Sun, X. Liu and H. Meng, "Improved End-to-End Dysarthric Speech Recognition via Meta-learning Based Model Re-initialization," *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, Hong Kong, 2021, pp. 1-5, doi: 10.1109/ISCSLP49672.2021.9362068.