Voice Recognition using mfcc and gmm

A PROJECT REPORT

Submitted by

Shubham Gupta
Reg. No. 18MCA1093

in partial fulfillment for the award of the degree of

Master of Computer Applications

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

School of Computer Science and Engineering
Vellore Institute of Technology - Chennai Campus
Vandalur - Kelambakkam Road, Chennai - 600 127

May - 2020

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

School of Computer Science and Engineering

DECLARATION

I hereby declare that the project entitled **Voice Recognition using mfcc and gmm** submitted by me to the School of Computer Science and Engineering, Vellore Institute of Technology - Chennai Campus, 600 127 in partial fulfillment of the requirements of the award of the degree of **Master of Computer Applications** is a bona-fide record of the work carried out by me under the supervision of **Prof.Thomas Abraham J V** . I further declare that the work reported in this project, has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or University.

Place: Chennai
Date:

Signature of Candidate
(Shubham Gupta)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled **Voice Recognition using mfcc and gmm** is prepared and submitted by **Shubham Gupta (Reg. No. 18MCA1093)** to Vellore Institute of Technology - Chennai Campus, in partial fulfillment of the requirement for the award of the degree of **Master of Computer Applications** is a bona-fide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Guide/Supervisor	**Head Of The Department**
Name: Prof.Thomas AbrahamJ V	Name: Dr M SIVAGAMI
Date:	Date:
Examiner	**Examiner**
Name:	Name:
Date:	Date:

(Seal of SCOPE)

Acknowledgement

I wish to express my sincere thanks to Dr. G.Viswanathan, Chancellor, for providing me an excellent academic environment and facilities for pursuing MCA program. I am grateful to Dr. R.Jagadeesh Kannan , Dean of School of Computing Science and Engineering, VIT Chennai. I wish to express my sincere gratitude to Dr. Sivagami M, Program chair of MCA for providing me an opportunity to do my project work. I would like to express my gratitude to my guide Prof.Thomas Abraham J V who inspite of his busy schedule guided me in the correct path. Finally, I would like to thank my family and friends who motivated me during the course of my project work

Shubham Gupta
Reg. No. 18MCA1093

Abstract

Recognition of speech is the ability to identify spoken words, and recognition of speakers is the ability to identify who is saying those words. Speaker Recognition is one of the main areas of research focused on Speech Signals. Speaking recognition, speech-to-text translation and vice-versa are other important field of study. In this paper, a brief overview of the area of speaker identification, describing its system, various modules for the extraction and modeling of features, applications, underlying techniques and some indications of performance is presented. The Mel Frequency Cepstral Coefficient (MFCC) is regarded as a key factor in the recognition of speakers. Gaussian Mixture Model (GMM) is the most popular model for data training. This considered MFCC as the primary feature with "tuned parameters," and delta- MFCC as a secondary feature and also implemented GMM with some tuned parameters to train our model. This shows that the role of identification of speakers can be performed using MFCC and GMM along with outstanding accuracy in the results of identification / diarization.

Contents

ListofFigures

Chapter 1

Introduction

Speaker Recognition is a person’s identification from speech features, voice biometrics which is also called voice recognition. There is a distinction between the recognition of the speaker(the recognition of who speaks) and the recognition of speech recognition of (what is said). These two words are often mixed and, in fact, voice recognition can be used for both. There is a difference between active authentication usually referred to as an authentication and identification of a speaker or a spoken voice, ultimately there is a difference between the speech recogni- tion and who talks and the diarisation of the speaker, recognizing when the same speaker is speaking. Speaker recognition could simplify the work of translating speech into systems trained in voices of particular persons, or it could be used as part of a safety process to authenticate or verify the speaker’s identity. Speaker acknowledgement has a history of four decades and uses the acoustically different features of speech. These acoustic patterns reflect anatomy, for instance the size, shape and composition of the throat and mouth. Speaking style for instance voice pitch. The verification of the speaker gained recognition by the speaker and is a behavioral, biometric classification. Where the speaker claims to be certain identity and the voices are used to confirm this claim, it is known as authentication or checking. On the other hand, identification is the task of evaluating and checking the unknown speaker identity as a 1:1 match in a sense. When a voice ismatched to a template, a voice printed or a voice model is also matched, whereas the voice identification is 1 and matches when comparing the voice to the templates in the safety perspective.
Automatic Speaker Recognition is the field of research that deals with digital signal processing associated with people’s identification based on their speech. Every person is unique in the shapes of his vocal tract, larynx sizes and other parts of his or her voice production organs. Apart from these physical differences, each individual speaker has its own characteristic way of speaking, linked to accent, rhythm, intonation style, pronunciation pattern, vocabulary selection. Aperson’s

voice is the most common technique of biometric recognition used to authenticate and track humans using their speech signal.
The aim of speaker recognition is to decide which speaker is present based on the individual’s pronunciation. Speaker recognition is the process for automati- cally identifying who is speaking by using the speaker-related information found in speech waves to check the identities claimed by people accessing systems. Speaker identity is related to physiological and behavioral characteristics of an individual speaker’s speech production system. The physiological module forthe recognition of the speaker is the physical shape of the vocal tract of the subject. The behavioral element is the physical movement of the lips, larynx and tongue.

History

In 1952 the first speech recognizer emerged and consisted of a device to iden- tify single spoken digits. Another early device was the IBM Shoebox, which was showcased at the New York World Fair in 1964. Lately there have been various developments on a single machine such as Sonic Extractor, such as a high speed mass transcription capability. Health care, and in particular the work of themedi- cal transcriptionist (MT), was one of the most important areas for the commercial application of speech recognition in the USA. According to industry experts, voice recognition (VR) was marketed at its introduction as a way to eradicate transcrip- tion entirely, rather than making the transcription process more effective, so it was not embraced. This was also the case that VR was still technologically in- adequate at the time. In addition, it demanded improvements to the way doctors worked and reported clinical experiences, which many, if not all, were unwill- ing to do, to be used effectively. Nevertheless, the main drawback to automating transcription voice recognition is regarded as the device. The essence of narrative dictation is highly interpretive and often involves judgment that can be provided by a real human being but not yet by an automated machine. Another drawback was the lengthy amount of time that the user and/or system provider took to train the program. A distinction is also made in Automatic Voice Recognition (AVR) between "artificial syntax systems" that are typically domain-specific and "natural language processing" that is typically linguistic specific. Each of these types of application presents specific goals and challenges of itsown.

Chapter 2

Overview/Literature Review

2.1 Overview

Biometric is unique to every individual physical feature. Owing to the growing number of applications for dialog systems, interest in this area has grown considerably in recent years. Nevertheless, the area of automatic speaker recognition has many open problems. The choice of suitable voice signal features and machine learning algorithms could be listed between them.
We also researched, compared and tried to combine various approaches and algorithms to find out which speaker recognition model is the most successful. We assume that the MFCC-GMM model is better suited based on parameters such as accuracy of detection, computation time, false rejection and false acceptance rate. The proposed method is a biometric variant of speech that integrates separately developed text speaker recognition.
When we were seeking the best possible solution to achieving our target, we tried and encountered many forums, journals, images, apps, libraries, etc. We're making only a few primary points here.

2.1.1 SPEAKER RECOGNITION

Speaker recognition involves practices that attempt to connect a sample of speech to its speaker by its acoustic properties. Speech signal is a multidimensional acoustic wave that deals with details about a speaker's features, spoken word, speaker's feelings, extra noise, channel transformations etc. Every single voice is a singular personal trait. The two individuals should have the same vocal system and equal synchronization of their articulators for indistinguishable speech which is least feasible. However, the speech exemplars produced from the same speaker often demonstrate a certain amount of difference.

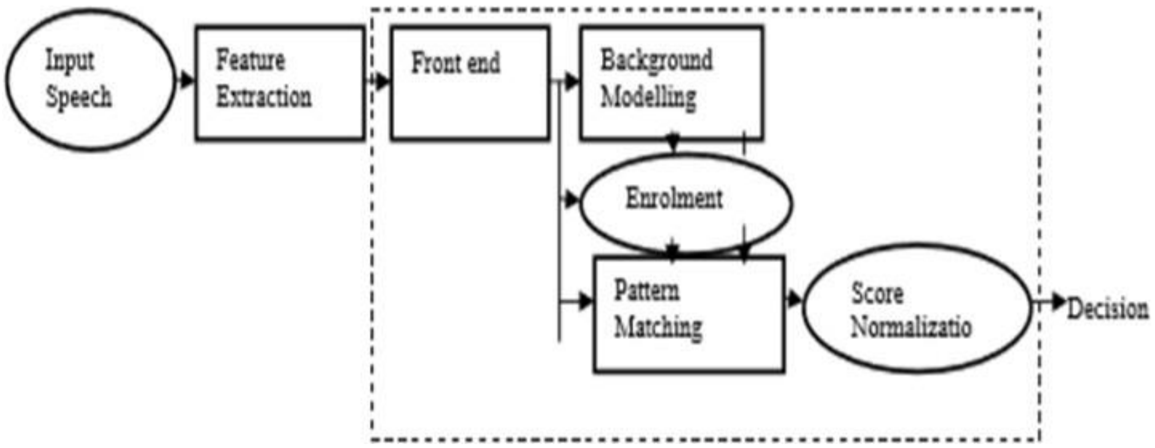


Figure 2.1: Speaker Recognition System

Intheidentificationofspeakerstheinitialstepistheextractionoffeatures. The raw signals are converted into feature vectors in the feature extraction module, in which speaker specific properties are emphasized and statistical redundanciesare concealed. A speaker model is trained in the enrolment mode using target speaker feature vectors. In recognition mode, system database is provided with feature vectors extracted from the person's utterance. A similarity value isgenerated.
The final decision is made on the basis of a similarity score by the decision module. A selection of background speakers or cohort speakers are used by nearly all state-of-the-art speaker recognition systems. It is done to improve the recognizer's robustness and computational efficiency. In the enrolment process, background speakers are used as negative examples in the training of adiscriminative model, or a universal background model in the training phase from which thetarget speaker models are adapted. In the reconnaissance phase background speakers are used to normalize the speaker matchscore.
It can be divided into Speaker Identification and Speaker Verification.

Speaker Identification

A speaker identification system only takes an unknown speaker's voice, and decides which speaker enrolled best fits the speech. The speaker identification system among the enrolled speakers finds the best matching speaker, and it may be that the unknown speaker is not enrolled. That is why the identification ofspeakers is accompanied by the speaker verification in manysystems. It's one-to-many comparisons (1:N match when the voice is compared to N templates). In the Speaker Identification System, the M speaker models are scored in parallel and most of the speakers are assigned an ID in the database, or none of the above will be reported if and only if the matching score is below a certain thresholdand is most likely to be reported in the case of an open speaker, and therefore the decision will be open-set Speaker Identification System.

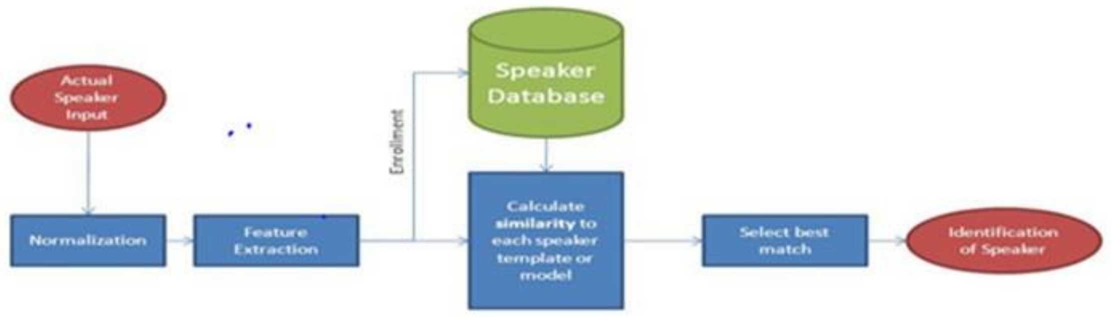


Figure 2.2: Speaker Identification Diagram

Speaker Verification

The verification of the speaker shall take the speech of an unknown speaker with his/her claimed identity and determine whether the claimed identity corresponds to the speech. Speaker tests are 1:1 when the voice of one speaker matches one template (also known as a "voice print" or "voice model") or, in other sense of the Pattern matches the claimed speaker model register in the database. If the match meets a certain threshold the claims for identity is confirmed. Using a high threshold, the system achieves high security and prevents the acceptance of impostors, but in the meanwhile it also risks rejecting the genuine person, and vice versa.

Speaker Text Dependent

A speaker recognition system may be fooled in a text-dependent mode by recording and playback of an enrolled speaker's predefined voice. The system will require the user to utter a randomly prompted text to protect a speaker recognition system against such malicious attack. In most cases, because additional information (text transcription) is given, a text based speaker recognition system outperforms a text Independent speaker recognition system. However, when the true underlying transcription is not given, a text-dependent speaker recognition system cannot be used, as in the situation when some are talking openly over the phone. Text-based identification works better for co-operating subjects.

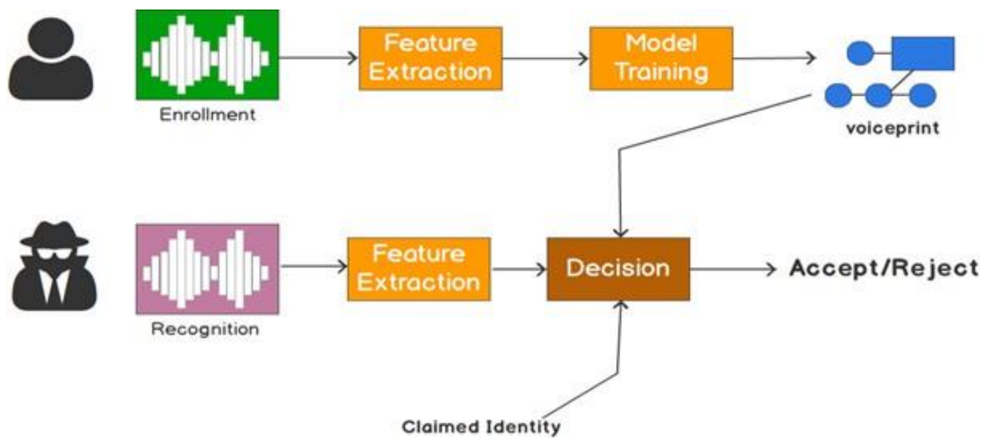


Figure 2.3: Speaker Verification Diagram

Speaker Text Independent

Text-independent systems are most often used to classify speakers, as they need very little if the speaker cooperates. In this case, the text is different during registration and testing In reality, registration is possible, as in various forensic applications, without the user's knowledge. Since text-independent technologies fail to compare what was said during registration and verification, verification applications often prefer to use speech recognition for evaluating, at the point of authentication, what the user says. Acoustics and voice analysis techniques are used in text-independent systems.

2.2 Literature Review

2.2.1 End-to-End Text-Dependent Speaker Verification by Georg Heigold

In this paper, author present a data-driven, integrated speaker verification approach, which maps a test utterance and a few reference statements directly to a single verification score and jointly optimizes the components of the method us- ing the same evaluation protocol and metric as at the test time. Such an approach would lead to clear, efficient systems requiring little domain-specific information and making little assumptions about the model. We implement the idea by formulating the problem as a single neural network architecture, including estimating a speaker model on just a few utterances, and testing it for text-dependent speaker verification on our internal "Ok Google" benchmark.

The proposed approach seems to be very successful for large data applications such as ours, which involve very precise, easy-to-maintain systems with a small footprint

D-Vector Baseline Approach

D-vectors are constructed from a deep neural network (DNN), representing an utterance as the speaker. A DNN consists of the successive application of several non-linear functions to turn the utterance of the speaker into a matrix in which a decision can be made easily. Portrays the topology of our DNN baseline. It comprises a local layer and several fully connected layers.
Except for the third, which is linear, all layers use ReLU Activation. During the training process, the DNN parameters are optimized using the softmax loss, which is defined for convenience as a linear transformation with

$$W_{spk}$$

weight vectors and biases

$$b_{spk}$$

, followed by the softmax function and the cross-entropy loss.

Experimental Evaluation

Theauthorevaluatetheproposedend-to-endapproachonourinternal”OkGoogle” benchmark.

- Data Sets and Basic setup.
- Frame-Level vs. Utterance-Level Representation
- Softmax vs. End-to-End Loss
- Feedforward vs. Recurrent Neural Networks

The author suggested a novel end-to-end approach to speaker verification, which maps the utterance directly to a score and optimizes both the internal speaker representation and the speaker model using the same loss for training and evaluation. Assuming adequate training data, on our internal "Ok Google" test, the proposed approach improved our best small footprint DNN baseline from over 3% to 2% equal error rate. Most of the gain came from modeling at the utterance- level versus frame-level.
The end-to-end loss obtained the same or slightly better performance compared to

other losses but with less additional con-cepts. For example, in softmax, we only obtained comparative error levels when using score normalization at runtime, candidate sampling to make workouts feasible, and dropout in training. In addition, we demonstrated that the same error rate can be further reduced to 1.4 percent using a recurrent neural network in place of a simple deep neural network, albeit at higher com-positional runtime expense. By comparison, a fair but not fully state-of-the-art i-vector / PLDA system gave 4.7%. There obviously needs to be more comparative studies. However, we believe our approach demonstrates a promising new path for Big Data Verification applications.

2.2.2 Deep Speaker: an End-to-End Neural Speaker Embedding System By Chao Li, Xiaokong Ma, Bing Jiang, Xiangan

Author present Deep Speaker, a system of neural speaker integration that maps utterances to an region where similarity between the speaker and cosine is measured. The built-in systems created by Deep Speaker can be used to recognize, check, and cluster speakers for several tasks. In ResCNN and GRU architectures, we play with the acoustic characteristics, and use three-fold losses based on the cosine similarity to mean pool for the produce utterance levelspeakers.

Trials of three different data sets show that a DNN based I base line is generated by Deep Speaker. For example, the Deep Speaker reduces the verification equivalent error rate by 50% (relatively) and increases the identification exactness of a text-independent dataset by 60% (relatively). We also present results that indicate that the adaptation to a Mandarin model will increase the accuracy of the recognition of English speakers.

A novel embedding scheme for end-to-end speakers, called Deep Speakers. The proposed method learns a mapping directly from speaker pronouncements to a hypersphere where cosine similarities correspond directly to a similar speaker scale. To obtain the acoustic features of the frame-level, we experiment with two separate neural network architectures (ResCNN and GRU). For metric learning is proposed a triple loss layer based on cosine similarities, along with a batch-global negative selection through GPUs. Pre-training Softmax is used to achieve greater efficiency.

The experiments show that, compared to the standard DNN-based I approach, the Deep Speaker algorithm greatly enhances the text-independent speaker recognition system. The EER decreases by around 50 percent in the Mandarin dataset UIDs, and error decreases by 60 percent. The same error rate decreases by 30% in the English MTurk dataset, and error decreases by 50%. Another strength of Deep Speaker is that for text-independent and text-dependent activities, it can take full advantage of transfer learning to solve speaker recognition problems on small data sets.

2.2.3 DEEP CNN BASED FEATURE EXTRACTOR FOR TEXT-PROMPTED SPEAKER RECOGNITION By Sergey Novoselov,Oleg Kudashev

Deep learning is not yet a very popular tool in the field of voice verification. In the task of text-prompted speaker verification, we study profoundly convolution- ary neural network performance. To check each digit utterance separately, the requested passphrase is segmented into word states — i.e. digits — They train a single high-level extractor of features for all states and use co-sine similarity metric to score. The main feature of our network is the activation function Max-Feature-Map which acts as an embedded selector of features.

By using the multitask learning scheme to train the high-level feature extractor, we were able to surpass the classic baseline systems in terms of quality and achieve impressive results for such a novice approach, resulting in the RSR2015 evalu-ation set at 2.85 percent EER. This result is strengthened by the merger of the proposed and Baseline systems.

Analyze the output of the proposed training data system extended to English (Wells Fargo Bank) and Russian (STC-Russian-Subcorporate) digits.

Features

CNN-based system also operates with separate states, and the same HMM-based segmentation is used to divide passphrases into separate digits. As features of input we use log mel power spectra extracted from the voice signal. While the feature size along the frequency domain axis is set at 64 bands, it is necessary to adjust the varying utterance time span to a fixed length of 96 frames, which is an estimate of the longest digit pronouncement time. It is achieved either by cropping the end of the characteristics along the time axis or by wrapping padding as achieved in. Standardization of the Cepstral mean and variation is achieved for every digit.

GMM-SVM

The GMM-SVM baseline system is implemented according to. A relevant max-imum a posteriori (MAP) adaptation of a passphrase's speaker-independent uni-versal background model (UBM) is obtained with the GMM mean supervector m~. This system does not affect segmentation. We trained a passphrase UBM on the devel-opment set of the RSR2015 database. Nui-sance Attribute Projection

(NAP) is applied to account for the variability of the interspeakers. Support Vector Machine (SVM) serves as a classifier for the backend. Rate s-normalisation is finally done.

This paper suggested a new deep CNN-based approach to the issue of text-pronounced speaker verification. A high level feature extractor has been equipped to distinguish simultaneously between speakers and digits, and embedding can be calculated simply with cosine-like similarity. By using the multitask learn- ing scheme, we were able to outperform the classic baseline systems in terms of consistency and obtain remarkable results for such a inexperienced approach, achieving 2.85 percent EER on the RSR2015 assessment package. The fusion of the proposed and baseline systems resulted in an EER decrease of nearly 50 per cent for all test sets.

2.2.4 Neural Voice Cloning with a Few Samples By Sercan Ö. Arık, Jitong Chen, Kainan Peng

Voice cloning is a widely sought after feature for personalized speech interfaces. Implement a neural voice cloning method, which learns from only a few audio samples to synthesize a person's voice. Researching two approaches: Adapta- tion of speakers and encoding of speakers. Adaptation of the speakers is based on the fine tuning of a generative multi-speaker configuration. Speaker encoding is based on training a separate model to explicitly infer an embedding of a new speaker that will be added to a generative multi-speaker model. Also with a few cloning audios, both methods will achieve strong performance in terms of speech naturalness and resemblance to the original speaker.
Although adaptation of speakers can achieve a slightly better naturalness and similarity, the cloning time and the required memory for the speaker encoding method are significantly lower, making it more suitable for deployment of low resources. This paper explores voice cloning in neural speech synthesis sequence-to-sequence systems. Our contributions include:

- We illustrate and evaluate the strength of speech adaptation approaches for voice cloning, based on a pre-trained multi-speaker model being fine-tuned for an unknown speaker using a few samples.
- We suggest a novel encoding method for speakers that offers comparable naturalness and consistency in subjective assessments while providing substantially less cloning time and requirements for computational resources.
- We suggest automated voice cloning evaluation methods based on classification of the neural speaker and speaker verification.

- Via embedding manipulations we illustrate voice morphing for gender and accent transformation.

Speaker adaptation

The principle of speaker adaptation is to use a few audio-text pairs to fine-tune a qualified multi-voice model to an unknown voice. The fine tuning can be applied either to the embedding speaker or to the entire platform. While the whole model provides more degrees of freedom to adapt speakers, its optimization is challenging for limited amounts of cloning data. It takes early stoppage to prevent over fitting.

Speaker encoding

They suggest a speaker encoding method to infer the speaker being inserted directly from an unknown speaker's audio samples. During voice cloning such a model does not require any fine-tuning. And the same configuration can be used for all the speakers that are not used.
For speaker encoders, we propose a three-part neural network architecture:

1. **Spectral processing:** We input mel-spectrograms of audio cloning samples into the prenet containing completely connected layers with an exponential linear unit for transformation of features.

2. **Temporary processing:** We use convolutionary layers with gated linear unit and residual connections to use the long-term sense, average pooling is applied to summarize the entire utterance.

3. **Cloning sample attention:** Given that different cloning audios contain different amounts of speaker information, we use a multi-head self-attention method to measure the weights for different audios and get aggregated embedding.

Speaker encoding

Spokesman classifier decides which speaker belongs to an audio sample. A speaker classifier with the collection of speakers used for cloning is qualified for voice cloning evaluation. High-quality voice cloning will result in high precision classification. Before the softmax feature, the architecture consists of separate spectral and temporal processing layers, and an additional embedding layer.

Speaker Verification

Speakerverificationisthejobofauthenticatingaspeaker'sallegedidentity,based on a test audio and enrolled speaker audios. In particular, binary classification is conducted to determine if the test audio and enrolled audios are from the same speaker. We find a verification model for the end-to - end text-independent speakers.

On a multi-speaker dataset, the speaker verification model can be trained and then used to verify whether the cloned audio and ground-truth audio are from the same speaker. Unlike the classification approach for speakers, speaker veri- fication model does not require cloning training from the target speaker with the audios, so it can be used with a few samples for unseen speakers. The corre- sponding error-rate (EER) 6 can be used as the quantitative performance metric to calculate how similar the cloned audios are to the audios of the ground reality.

They research two methods for cloning of neural voices: adaptation of speakers and encoding of speakers. Demonstrate that even with just a few cloning au- dios, both methods will achieve reasonable cloning efficiency. For naturalness, we show that adaptation of speakers as well as encoding of speakers can achieve a MOS close to the multi-speaker generative model of the baseline. Therefore, the strategies suggested could theoretically be improved in the future with better multi-speaker models (such as replacing Griffin-Lim with WaveNet vocoder). For comparison, we show that both methods benefit from a greater number of audios being cloned.

The output difference between full-model and embedding-only adaptation suggests that, in addition to speech embedding, other discriminative speaker knowledge still exists in the generative model. Compact representation through embeddingbenefitsfromquickcloningandlowfootprintperspeaker. Wefinddrawbacks of training the multi-speaker generative model using a voice recognition dataset with poor quality audios and restricted diversity of speakers. Improvements in dataset consistency will result in a greater naturalness. Weexpect our techniques to benefit greatly from a multi-speaker dataset of great and highquality.

2.2.5 Gender Recognition by Voice Using an Improved Self-Labeled Algorithm by Ioannis E. Livieris, Emmanuel Pintelas, Panagiotis Pintelas

Speech recognition has various applications including human to machine contact, sorting by gender categorization of telephone calls, marking videocategorization and so on. Machine learning is currently a common phenomenon that has been widely used in different fields and applications, leveraging the rapid growth of digital technology and taking advantage of electronic media storage capabilities.

Recently, research focuses on integrating learning methods for the ensemble with the semi-supervised learning system aimed at creating more accurate classifiers. In this paper, concentrate on gender identification by voice using a new self- labeled, semi-supervised ensemble algorithm. Our preliminary numerical experi- ments demonstrate the classification efficiency of the proposed algorithm in terms of precision, leading to stable and robust predictive models beingcreated.

On Semi-supervised Self-Labeled Classification Algorithms

The self-labeled methods are divided into two main categories, Self-training and Co-training.

[1] Self-training

In self-training, one classifier is initially trained on a labeled training set and used to predict unlabeled set examples where the most positive predictions areapplied to the labeled training set. Then, the classifier is retrained on the new expanded label set and this procedure is repeated until some stop conditions, such as not having unlabeled examples left, have been met. The benefit of the Self-training algorithm is its simplicity, being one of the most efficient self-labeledalgorithms at the same time. One drawback, however, is the potential inclusion of examples of noise into the labelled trainingset.

[2] Co-training

Co-training is a multi-view algorithm which means that the space of the function is divided into two independent views, conditionally. Two classifiers are initially trained separately on each view using the labeled set, and each classifier then applies the most confident predictions to the other's training set using an itera- tive procedure. These two algorithms influence most self-labeled methods while some of them are also based on ensemble techniques. SETRED, Tri-Training, Co-Forest, and Co-Bagging are some of the other effective self-labeled algorithms proposed in the literature.

[3] Democratic-Co learning

Democratic-co-learning is a single-view algorithm that uses multiple classifiers to predict the values of unlabeled examples that implement a majority voting strat- egy and a mean measure of trust for the majority and minority of disagreeable classifiers.

[4] SETRED

SETRED is a self-labeled algorithm that integrates data editing into the self-training framework to actively learn from the self-labeled examples. More analytically, data editing is a tool that improves the consistency of the training set by defining and deleting the mislabeled examples obtained from a self-study procedure using some local information in a graph of theneighborhood.

[5] Tri-training

The Tri-training algorithm is a single view algorithm that uses three classifiersto teach each other on the basis of a majority voting strategy, thereby avoiding the trust calculation of the classifier's labeling because this procedure is often time-consuming and complicated.
In particular, the initial training set trains three classifiers on data subsets generated by bootstrap sampling, which are then used to label the instances of the unlabeled set. If two classifiers agree to predict a value for an unlabeled example, then the third is labeled as well.

[6] Co-Forest algorithm

This algorithm uses random trees, which are trained from the dataset on bootstrap data allocating a few unlabeled examples to each random tree. A plurality vote is used to make the final decision. Two advantages compared to the remaining self-labeled algorithms are that due to the random selection of features from the basic feature vector and the reduced ripples of its performance in the case of a small number of labeled examples given, no physical connection between theattributes is required.

[7] Co-Bagging algorithm

The Co-Bagging algorithm uses many base classifiers trained on bootstrap data generated by random resampling with training set replacement. Every sample comprises approximately two thirds of the training package. One advantage is that this algorithm works well for unstable learning algorithms where a small change in inputtrainingdatacancauseabigchangeinoutputhypothesistobesignificant.

Conclusions

In this work, we used a semi-supervised voice-recognition algorithm called iCST-Voting. The proposed algorithm is an ensemble of the most common self-labeled algorithms, i.e. , self-training, co-training and tri-training, using an ensemble of classifiers as a base learner. Compared to other similar approaches, the contribution of our approach has to do with the fact that we use a classifier ensemble as a base learner rather than single learners that are usually used in self-labeled algorithms.

Our preliminary numerical results and the statistical analysis presented show the effectiveness of the proposed algorithm for voice-recognition of gender compared to self-labeled state-of-the-art. It also presents competitive and, at times, better classification performance than conventional supervised algorithms. We therefore conclude that accurate, stable, and robust prediction models could be established in the semi-supervised learning system through the adaptation of ensemble techniques.

Chapter3

SystemDesign

Speaker recognition involves practices that attempt to connect a sample of speech to its speaker by its acoustic properties. Speech signal is a multidimensional acoustic wave that deals with details about a speaker's features, spoken word, speaker's feelings, extra noise, channel transformations etc. Every single voiceis a singular personal trait. The two individuals should have the same vocal system and equal synchronization of their articulators for indistinguishable speech which is least feasible. However, the speech exemplars produced from the same speaker often demonstrate a certain amount of difference.

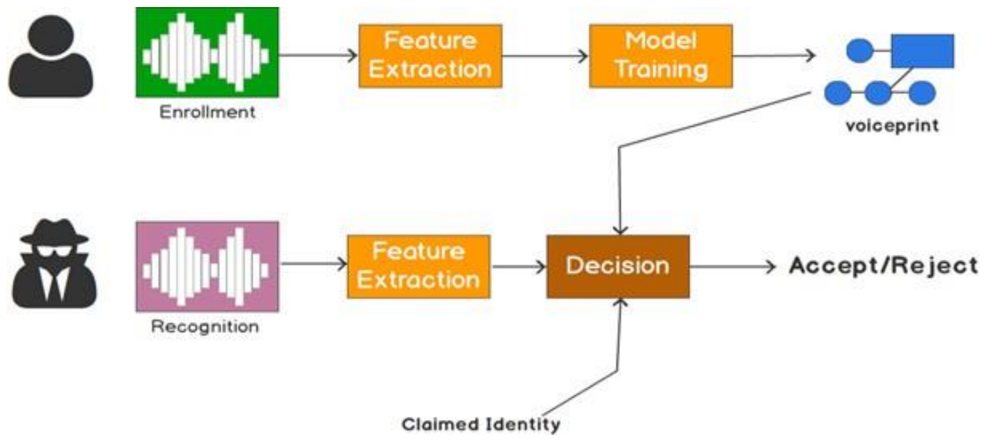


Figure 3.1: Speaker Diagram

3.1 Enrollment in Speaker Recognition

Enrolling a user into a speech recognition system involves analyzing a user utterance's acoustic content and determining whether the utterance matches a portion of an enrollment text. If a part of the enrollment text fits the user utterance, the acoustic contents will be used to update acoustic models for the part of the enrollment text. A speech recognition system analyzes a user's speech to determine what the user said. Most speech recognition systems are frame-based. In a frame-based system, a processor divides a signal descriptive of the speech to be recognized into a series of digital frames, each of which corresponds to a small time increment of the speech.

A speech recognition system can be a "discrete" system recognizing discrete words or phrases but allowing the user to pause momentarily between each discrete word or phrase. Alternatively, a speech recognition program may be a "continuous" de- vice capable of recognizing spoken words or phrases regardless of whether the user pauses. Continuous speech recognition systems usually have a higher inci- dence of recognition errors compared to discrete recognition systems because of the limitations of continuous speech recognition.

The innovation features enrolling the user in a speech recognition program by evaluating the acoustic quality of the user's utterance and, based on an analysis, deciding if the user's utterance corresponds to a portion of the enrollment text. The user utterance's acoustic content is used to update the acoustic models that correspond to the enrollment text section if the user utterance matches a section of the enrollment text.Can pick the enrollment text from a plurality of enrollment texts. Every enrollment text has a grammar corresponding to its enrollment. To decide if the user utterance fits a portion of the enrollment text, the enrollment grammar corresponding to the selected enrollment text is used.A user canreceive a text of the enrollment. For the purpose of determining whether the user utterance matches a portion of the enrollment text, an enrollment grammar corresponding to the received enrollment text may be created.

When a part of the enrollment text does not suit, the user utterance may be ignored. Certain implementations of the invention may include one or more of the following features. An enrollment grammar corresponding to the enrollment text may be used to determine whether the user utterance matches a portion of the enrollment text. A rejection grammar may be used to determine whether the user utterance matches a portion of the enrollment text. The rejection grammar may be a phoneme grammar and may model an utterance using a set of phonemes that is smaller than a set of phonemes used by the enrollmentgrammar.

In general, the innovation features another aspect of enrolling a user into a speech recognition program by showing an enrollment text and an enrollment location within the enrollment text. When receiving a user utterance a decision is made

as to whether there is a connection between the user utterance and a portion of the enrollment text beginning at the enrollment location. When a match occurs, the enrollment location will be changed, and the new enrollment position will be shown.
Certain implementations of the invention may include one or more of thefollowing features. The enrollment position may be displayed using a marker at the enrollment position. The enrollment position may be displayed by highlighting the enrollment text at the enrollment position, and may be displayed using a cursor at the enrollment position. the invention features a speech recognition system for enrolling a user. The system includes a display for displaying an enrollment text to a user, an input device for receiving speech signals, and a processor. Theprocessor determines a user utterance from a received speech signal, analyzes acoustic content of the user utterance, and determines, based on the acoustic analysis, whether the user utterance matches a portion of an enrollment text. The processor then uses the user utterance to update acoustic models corresponding to the portion of the enrollment text if the user utterance matches a portion of the enrollment text. Among the advantages of the invention is that by determining and ignoring user utterances that do not match a portion of the enrollment text, the acoustic models oftheenrollmentprogramarenotincorrectlyupdatedbasedontheuserutterance.

3.2 FeatureExtraction

The extraction of features is a reduction of dimensionality by which an initial set of raw data is reduced to more manageable groups for processing. A feature of these big data sets is a large number of variables that require a lot of computing resources to process. Extraction of features is the term for methods which select and/or combine variables into features, effectively reducing the amount of data to be processed while still accurately and completely describing the original set of data.
The extraction of feature process is useful when reducing the number of resources needed to process without losing important or relevant information. Extraction of features may also reduce the amount of redundant data needed for a given analysis. In addition, data reduction and the machine’s efforts to create variable combinations (features) accelerate the learning speed and generalization steps in the machine learning process.
Extraction of features involves reducing the number of resources needed to describe a broad set of data. One of the major problems arises from the number of variables involved when performing an analysis of complex data. Analysis with a large number of variables usually requires a large amount of memory and computational resources, which can also cause a classification algorithm to over fit

samples for training which generalize poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still representing the data with adequate accu- racy. Many machine learning practitioners assume that properly designed feature extraction is the secret to successful modelconstruction.

3.2.1 Mel Frequency Cepstral Coefficient

MFCC are the Mel Frequency Cepstral Coefficients. MFCC takes into account human sensitivity experience at appropriate frequencies by converting the standard scale to Mel Scale, and is thus very suitable for speech recognition tasks (as they are suitable for human perception and the frequency at which humans speak/utter).

Features extraction and recognition are two essential modules in speech recognition systems. The primary aim of extraction of the function is to identify robust and discriminatory features in the acoustic data. The recognition module decodes the speech input using speech features and acoustic models and generates textresults with high precision. The main purpose of the extraction of feature step is to compute a saving sequence of feature vectors that provide a compact representation of the given input signal. First of all, the recording of different speech samples of each word in the vocabulary is done by different speakers. After the speech samples are collected; sampling at a frequency of 20 kHz converts them from analog to digital form. Sampling requires the recording at a regular inter- val of the speech signals. The data collected are now quantified if it is necessary to eliminate noise in speech samples. Speech samples collected are then passed through the extraction feature, training feature & testing stage. Feature extraction transforms the incoming sound into an internal representation so that the original signal can be reconstructed from it. MFCC also increasingly finds uses in music information such as gender classification, audio similarity measurement, etc.The basic concept of the MFCC method is shown in the block diagram ofFig.

This is where we concentrate on two key features of MFCCs and their derivatives, Delta-MFCC states. Wehad 20 MFCCs and 20 Delta-MFCCs estimated. So, we had 40 apps totally in hand. Under featureextrction.py module, Delta MFCC was calculated with a custom definedfunction.

MFCC is one of the essential feature extraction techniques for speaker identification techniques. The purpose of feature extraction is to find a collection of utterance properties that have acoustic voice signal correlations that are parameters that can somehow be measured or estimated by analyzing the waveform of the signal. These parameters are called features.

When dealing with the speech signal, the extracted features will have certain parameters, such as:

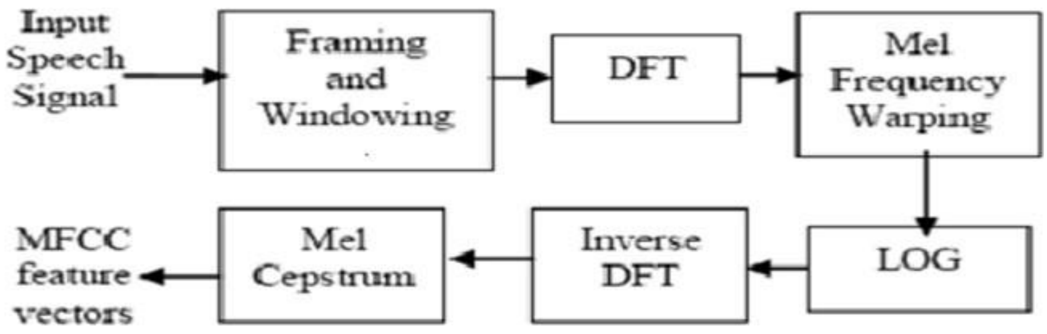


Figure 3.2: Steps involved in MFCC Feature Extraction

- Stable over Time
- Should occur frequently and naturally in speech
- Should not be susceptible to mimicry
- Easy to measure extracted speech features
- Shows little fluctuation from one speech to another environment
- Discriminate between speakers while being tolerant of the variability ofin-tra speakers.

Pre-emphasis

Pre-emphasis is important because voiced parts of the speech signal naturally have a negative spectral slope (attenuation) of around 20db per decade because of the physiological characteristics of speech development. Pre-emphasis raises high frequency energy level. For voiced segments, such as vowels, the lower frequen-cies have more energy than the higher frequencies. This is called spectral tilt (how vocal folds produce sound) which is related to the glottal source. The boost- ing of the high frequency energy makes the acoustic model more accessible to information in higher formants. For humans, we're starting to have hearing prob-lems when we can't hear these high-frequency sounds. Also, noise has a high frequency. We're using pre-emphasis to make the system less noise-sensitive later in the process. For some applications, we just need to undo the boost at the end. Pre-emphasis uses a filter to boost higher frequencies. Below is the before and after signal to boost the high-frequency signal.

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2}$$

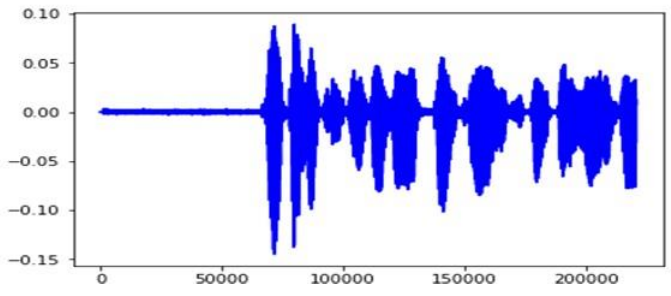


Figure 3.3: Signal in the Time domain before Pre-emphasis

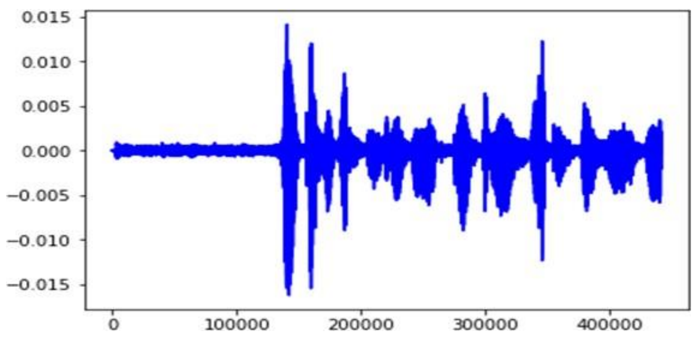
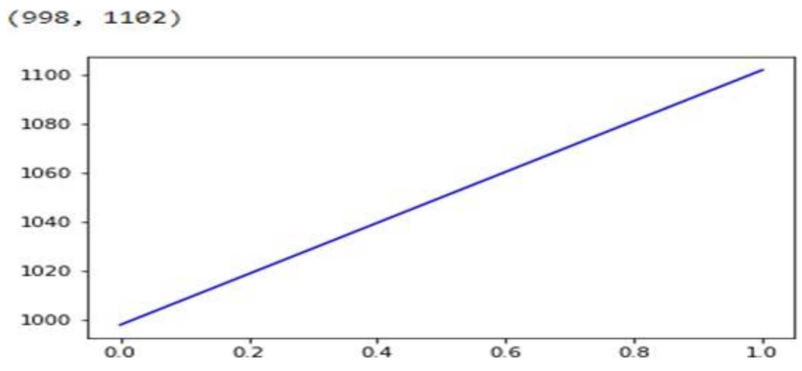


Figure 3.4: Signal in the Time domain after Pre-emphasis

Framing

We need to split the signal into short-time frames, after pre-emphasis. The explanation for this step is that frequencies in a signal change over time, so in most situations, there's no point in transforming the Fourier throughout the whole signal by losing the signal's frequency contours over time. To prevent this, we can safely assume that the frequencies in a signal are stationary over a very short time period. Thus, by making a Fourier transformation over this short-term frame, we can achieve a good approximation of the frequency contours of the signal by concatenating adjacent frames.

Typical frame sizes range from 20 ms to 40 ms in speech processing, with 50 percent (+ /-10 percent) overlapping between consecutive frames. For the frame size, the common settings are 25 ms, frame size = 0.025 and 10 ms stride (15 ms overlap), frame stride = 0.01.

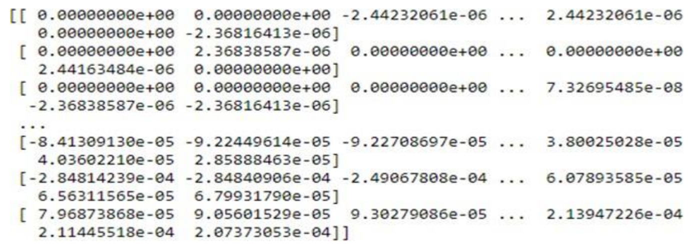


```
(998, 1102)
```

Figure 3.5: Frame shape and Plot

Windowing

After the signal is sliced into frames, we add to each frame a window function such as the Hamming window. The next step in processing is to window each frame to minimize signal discontinuities at the beginning and end of each frame. The idea is to reduce the spectral distortion by tapering the signal at the beginning and end of each frame via the window. If the window is defined as w(n), 0 <= n <= N-1, where N is the number of samples in each frame, then the result of the signal window is y(n) = x(n) * w(n) where x(n) is the speech signal beingprocessed.



```
[[ 0.00000000e+00  0.00000000e+00 -2.44232061e-06 ...  2.44232061e-06
   0.00000000e+00 -2.36816413e-06]
 [ 0.00000000e+00  2.36838587e-06  0.00000000e+00 ...  0.00000000e+00
   2.44163484e-06  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00 ...  7.32695485e-08
  -2.36838587e-06 -2.36816413e-06]
 ...
 [-8.41309130e-05 -9.22449614e-05 -9.22708697e-05 ...  3.80025028e-05
   4.03602210e-05  2.85888463e-05]
 [-2.84814239e-04 -2.84840906e-04 -2.49067808e-04 ...  6.07893585e-05
   6.56311565e-05  6.79931790e-05]
 [ 7.96873868e-05  9.05601529e-05  9.30279086e-05 ...  2.13947226e-04
   2.11445518e-04  2.07373053e-04]]
```

Figure 3.6: Features of Window

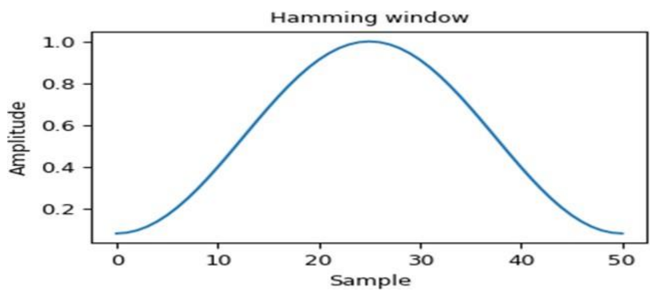
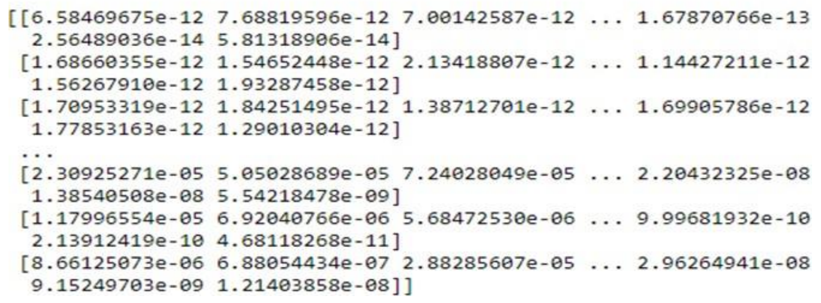


Figure 3.7: Hamming Window

Fourier-Transform and Power Spectrum

We now can perform N-point FFT on each frame to compute the frequency spectrum, called Short time Fourier-Transform (STFT), with N usually 256 or 512, NFFT = 512, and then use the following equation to measure the power spectrum (periodogram).

$$P=|FFT(x_i)|_2N$$



```
[[6.58469675e-12 7.68819596e-12 7.00142587e-12 ... 1.67870766e-13
  2.56489036e-14 5.81318906e-14]
 [1.68660355e-12 1.54652448e-12 2.13418807e-12 ... 1.14427211e-12
  1.56267910e-12 1.93287458e-12]
 [1.70953319e-12 1.84251495e-12 1.38712701e-12 ... 1.69905786e-12
  1.77853163e-12 1.29010304e-12]
 ...
 [2.30925271e-05 5.05028689e-05 7.24028049e-05 ... 2.20432325e-08
  1.38540508e-08 5.54218478e-09]
 [1.17996554e-05 6.92040766e-06 5.68472530e-06 ... 9.99681932e-10
  2.13912419e-10 4.68118268e-11]
 [8.66125073e-06 6.88054434e-07 2.88285607e-05 ... 2.96264941e-08
  9.15249703e-09 1.21403858e-08]]
```

Figure 3.8: Features of NFFT

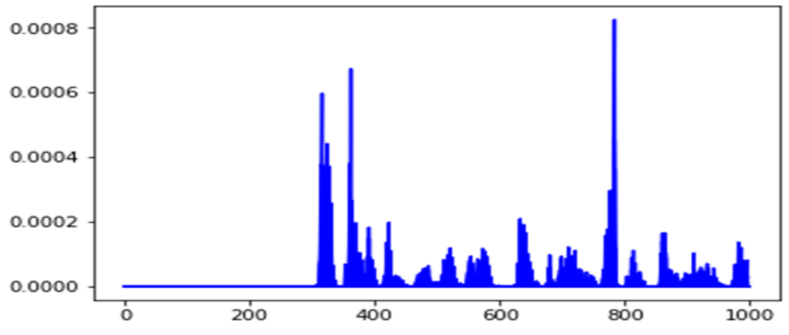
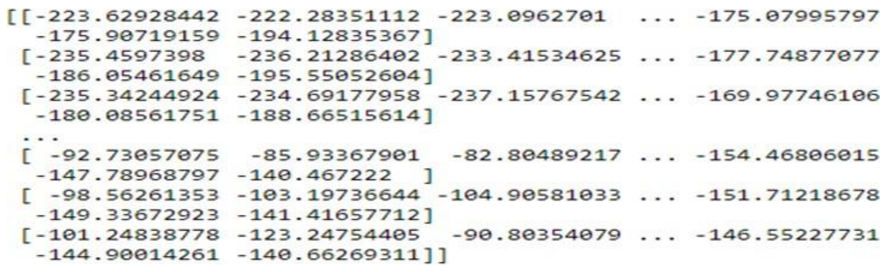


Figure 3.9: Plot of NFFT

Filter Banks

The final step towards computing filter banks is to add triangular filters, usually 40 filters, nfilt = 40 to the power spectrum on a Mel-scale to obtain frequency bands. The Mel-scale attempts to imitate the non-linear experience of sound in the human ear by being more selective at lower frequencies and less discriminatory at higher frequencies. The filter is triangular in the center of the filter bank and has a response of 1 until it hits the middle of two adjacent filters, with a response of 0.



```
[[-223.62928442 -222.28351112 -223.0962701   ... -175.07995797
  -175.90719159 -194.12835367]
 [-235.4597398   -236.21286402 -233.41534625 ... -177.74877077
  -186.05461649 -195.55052604]
 [-235.34244924 -234.69177958 -237.15767542 ... -169.97746106
  -180.08561751 -188.66515614]
 ...
 [ -92.73057075  -85.93367901  -82.80489217 ... -154.46806015
  -147.78968797 -140.467222  ]
 [ -98.56261353 -103.19736644 -104.90581033 ... -151.71218678
  -149.33672923 -141.41657712]
 [-101.24838778 -123.24754405  -90.80354079 ... -146.55227731
  -144.90014261 -140.66269311]]
```

Figure 3.10: Extracted Filter Banks

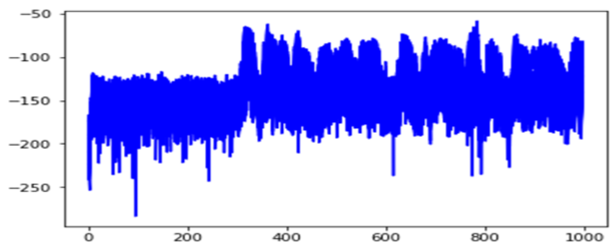


Figure 3.11: Plot of Filter Banks

3.2.2 Python speechfeatures

This library provides common speech features for ASR including MFCCs and filterbank energies. You will need numpy and scipy to run these files.

Supported features

- python_ speech_ features.mfcc() - Mel Frequency CepstralCoefficients
- python_ speech_ features.fbank() - Filterbank Energies
- python_ speech_ features.logfbank() - Log Filterbank Energies
- python_ speech_ features.ssc() - Spectral Subband Centroids

We used this particular library as part of our project. It satisfied all of our feature engineering requirements without causing any trouble. In identification task we even achieved remarkable accuracy.

Supported features

python_ speech_ features.base.mfcc: (signal, samplerate=16000, winlen=0.025, winstep=0.01, numcep=13, nfilt=26, nfft=512, lowfreq=0, highfreq=none, pre-emph=0.97, ceplifter=22, appendEnergy=True, winfunc=<function <lambda»)

Parameters:

- signal–theaudiosignalfromwhichtocomputefeatures. ShouldbeanN*1 array
- samplerate – the samplerate of the signal we are workingwith.
- winlen – the length of the analysis window in seconds. Default is 0.025s (25 milliseconds)
- winstep–thestepbetweensuccessivewindowsinseconds. Defaultis0.01s (10 milliseconds)
- numcep – the number of cepstrum to return, default 13
- nfilt – the number of filters in the filterbank, default26.
- nfft – the FFT size. Default is 512.
- lowfreq – lowest band edge of mel filters. In Hz, default is 0.
- highfreq – highest band edge of mel filters. In Hz, default is samplerate/2
- preemph – apply preemphasis filter with preemph as coefficient. 0 is no filter. Default is 0.97.
- ceplifter – apply a lifter to final cepstral coefficients. 0 is no lifter. Default is 22.
- appendEnergy–ifthisistrue,thezerothcepstralcoefficientisreplacedwith the log of the total frame energy.
- winfunc–analysiswindowtoapplytoeachframe. Bydefaultnowindowis applied. Use numpy window functions heree.g.winfunc=numpy.hamming

Returns:

A numpy array of size (NUMFRAMES by numcep) containing features. Each row holds 1 feature vector.

Chapter 4

Implementation of System/ Methodology

We eventually stuck to using python speech features library after breaching through numerous approaches and specific libraries. Our main methodology includes extraction of features via MFCC and then GMM Model Training. The image below pictorially demonstrates our methodology.

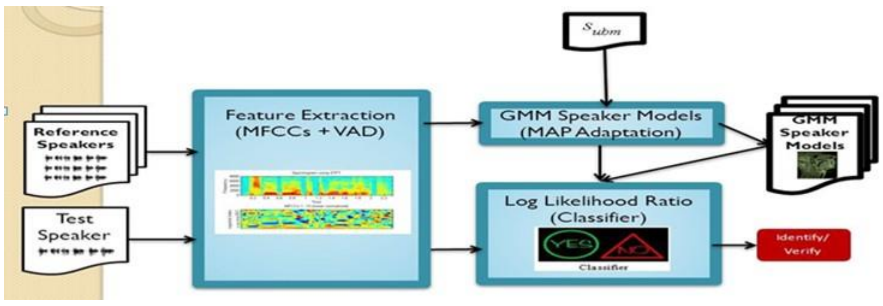

s_{ubm}
Reference Speakers
Test Speaker
Feature Extraction (MFCCs + VAD)
GMM Speaker Models (MAP Adaptation)
GMM Speaker Models
Log Likelihood Ratio (Classifier)
YES
NO
Classifier
Identify/ Verify

Figure 4.1: Flow of Speaker Recognition System

4.1 Model Training

We implemented the GMM approach for model training. The speaker identification system module can be separated into four modules:

- Front-end processing
- Speaker modeling

- Speaker database
- Decision logic

4.1.1 Front-end processing

This step represents the first step towards generating vectors for features. It is the "signal processing" component, which transforms the sampled speech signal into a collection of feature vectors, characterizing the speech properties that can distinguish different speakers. The front-end analysis is carried out in both the training and test phases. The purpose of the front-end processing is to adjust the speech signal to make it more suitable for extraction analyzes of features. The front-end processing procedure focused on the cancelation, framing, windowing and pre-emphasis of noise. The aim of feature extraction is to find a collection of utterance properties that have acoustic correlations to the speech-signal, that is, parameters that can be computed or estimated by processing the signal waveform in some way. These parameters are called functionalities. It involves the process of measuring some essential signal characteristics such as energy or frequency response, increasing these measurements with some perceptually relevant derived measurements and dynamically conditioning these numbers to form vectors for observation

4.1.2 Modeling

Modeling technique aims at creating models for each speaker using unique vector function extracted from each speaker. It performs function data reduction by modeling feature vector distributions. The reorganization of speakers is also split into two sections which means dependent speaker and independent speaker. In speech reorganization speaker independent mode, the device will disregard the speaker specific characteristics of the speech signal and extract the intended message. On the case of speaker-dependent speech reorganization system, on the other hand, speaker characteristics should be extracted in the acoustic signal. The main purpose of recognizing speakers is to compare a voice.
Modeling of the speakers requires two stages of preparation and research. Speaker models are produced using a particular characteristic of voice. There are two types of methods of model creation; stochastic models, and prototype models. Such modeling approaches use the features derived from the speech signal to create speaker models. In this step a voice model is generated and stored based on the extracted characteristics of the speech signal.
Matching algorithm compares the models of the stated user when authenticat- ing a speaker. In stochastic models, pattern matching is probabilistic, and the

consequence is probability estimation, or conditional probability, of the observer model being given. The prototype approach can be time based or independent. VQ modeling is an example of a prototype model which is independent of time. Time-dependent prototype model is more complex, because it must handle varia- tion in the speaking rate of humans. Stochastic models are more robust andresult is more accurate as compared to template models due to probabilistic probability ranking.

4.1.3 Gaussian Mixture Model

Gaussian mixture models are a probabilistic model for describing subpopulations normally distributed across an overall population. Generally, mixture models do not need to know which subpopulation a data point belongs to, allowing the model to automatically learn the subpopulations. Since the role of subpopulation is not recognized, this is a form of unsupervised learning. GMMs were used to extract features from speech data, and were also commonly used in object tracking of multiple objects, where the number of mixture components and their means predict object positions in a audio sequence at each frame. The concept of training a GMM is to approximate the distribution of probabilities in a class by means of a linear combination of the ’ k ’ Gaussian distribution-clusters, also called the GMM components. The probability of data points for a model (feature vectors) is given as

equation: $$P(X|\lambda) = \sum_{k=1}^{K} w_k P_k(X|\mu_k, \Sigma_k) \quad (1)$$

Where the Gaussian distribution is $P_k(X|\mu_k, \Sigma_k)$

$$P_k(X|\mu_k, \Sigma_k) = \frac{1}{\sqrt{2\pi|\Sigma_k|}} e^{\frac{1}{2}(X-\mu_k)^T \Sigma^{-1}(X-\mu_k)} \quad (2)$$

For the parameters mean μ, co-variance matrices Σ and weight w of the k elements, the training data of the class X_i is used.

The object Gaussian Mixture implements the expectation-maximization (EM) algorithm for fitting Gaussian models. It can also draw trust ellipsoids for multi-variate models, and measure the Bayesian Information Criterion for evaluating the number of clusters in the data. A GaussianMixture.fit method is provided which learns from train data about a Gaussian Mixture Model. Given the test data, it can allocate the Gaussian to each sample it most likely belongs to using the Gaussian-Mixture.predict process.

We use the sklearn.mixture package from Python to learn a GMM from the features matrix which contains the features of the MFCC. The GMM object requires the number of components n components to be fitted on the data, the number of iterations n_ iter to be performed to estimate the parameters of those n_ components, the type of co-variance covariance type to be assumed between the features and the number of times n_ init the initialization of K-means is to be performed. It retains the initialization that delivered the best results. The fit() function then uses the EM algorithm to estimate model parameters

Pros and Cons of Gaussian Mixture

[1] Pros

Speed: It is the fastest algorithm for learning mixture models.
Agnostic: As this algorithm maximizes only the likelihood, it will not bias the means towards zero, or bias the cluster sizes to have specific structures that might or might not apply.

[2] Cons

Singularities: It is difficult to estimate the covariance matrices when one has insufficiently many points per mixture, and the algorithm is known to diverge and to find solutions of infinite probability when one regularizes the covariances artificially.
Number of components: This algorithm will always use all the components to which it has access, need hold-out data or theoretical knowledge requirements to determine how many components to use in the absence of external references.

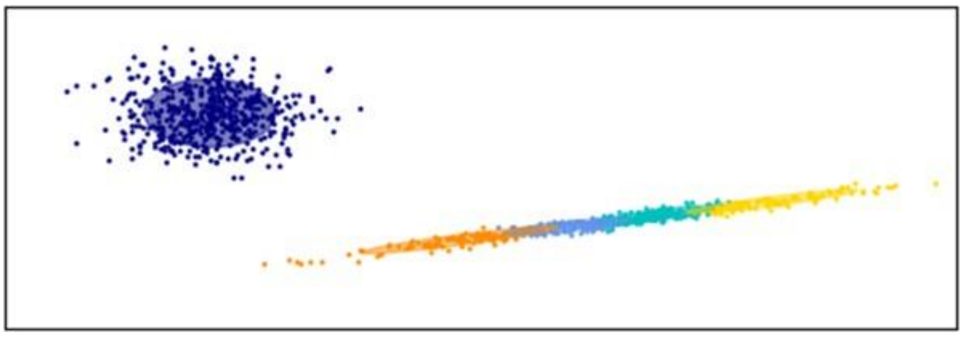


Figure 4.2: Gaussian Mixture

4.1.4 Speaker Modeling and Database Creation

Modeling of the speakers requires two stages of training and testing. Speaker models are created using a specific characteristic of voice. There are two types of

methods of model creation; stochastic models, and template models. These modeling methods use the features derived from the speech signal to create speaker models. During this step a voice model is generated and stored based on the extracted characteristics of the speech signal.
Matching algorithm compares the models of the stated user when authenticating a speaker. In stochastic models, pattern matching is probabilistic, and the consequence is probability estimation, or conditional probability, of the observer model being given. The prototype approach can be time based or independent. VQ modeling is an example of a prototype model which is independent of time. Time-dependent prototype model is more complicated because it must handle variation in the human speaking rate. Stochastic models are more robust and result is more accurate as compared to template models due to probabilistic probability ranking.

4.1.5 Features Matching

Matching is the process of comparing a individual with stored speaker models / templates with the extracted speech features. The comparison quantifies the similitude between a voice (identification record) and a speech database speaker model. Choosing a matching strategy is a crucial task. There are several common classifications / matching techniques such as the Gaussian MixtureModel.
Speaker model is used to interact with a given input signal. Models for the speak- ers are stored in a database. Two kinds of database are there: training database and testing database. Model comparison method involves the following:

- Matches training database of the target speaker with his/her testing database.
- Match score is calculated
- If the match score is greater than or equal to the threshold then the target speaker is accepted by the system otherwise rejected.

In the case of speaker recognition system, speaker-dependent and speaker-independent in the case of forensic speaker recognition system, can be gener- ated during matching speaker models. Relevant parameters for producing speaker models are predefined.

4.1.6 Decision Phase

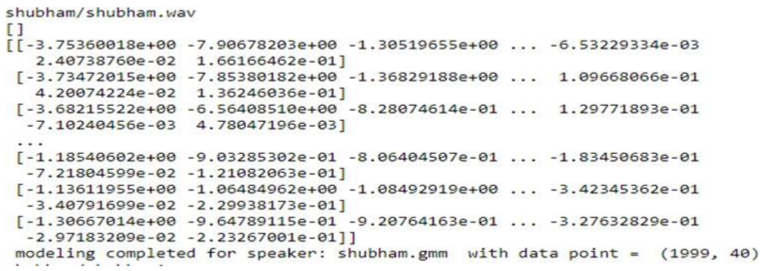
The decision process is based on the form of system , i.e. closed-set or open-set system. In the case of a closed-set recognition method, the decision can be made by choosing the one that is most similar to the voice signal of the test sample. In the case of open-set, program needs a threshold to check the validity of similarity.

As there could be chances that a device may reject a registered speaker, the cost of making an mistake in the decision process is also considered.

For example, in the case of a bank accepting an imposter, it would prove more costly than refusing a true customer. Relevant matching and simulation algorithms decide the Decision. For example, in case of a template matching decision, the calculated distance between speaker models is given whereas the calculated result in stochastic matching is based on the calculated probabilities. It makes the final decision about the identity of the speaker by comparing unknown speaker to all models in the data base and selecting the best matching model.

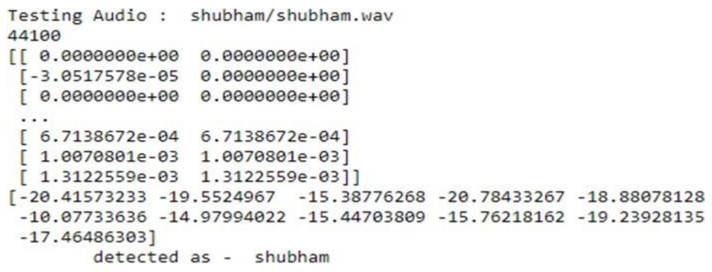
4.1.7 Performance evaluation phase

The framework also provides the means for evaluating speaker recognition system efficiency. Various accessible metrics may be used for that reason. The log-likelihood for each .gmm model of every speaker was calculated in the model training phase. It was stored as a database in a separate folder. This data dictionary is used for matching 1: N speaker's gmm file. The speaker with the highest score is chosen and identified.



```
shubham/shubham.wav
[]
[[-3.75360018e+00 -7.90678203e+00 -1.30519655e+00 ... -6.53229334e-03
   2.40738760e-02  1.66166462e-01]
 [-3.73472015e+00 -7.85380182e+00 -1.36829188e+00 ...  1.09668066e-01
   4.20074224e-02  1.36246036e-01]
 [-3.68215522e+00 -6.56408510e+00 -8.28074614e-01 ...  1.29771893e-01
  -7.10240456e-03  4.78047196e-03]
 ...
 [-1.18540602e+00 -9.03285302e-01 -8.06404507e-01 ... -1.83450683e-01
  -7.21804599e-02 -1.21082063e-01]
 [-1.13611955e+00 -1.06484962e+00 -1.08492919e+00 ... -3.42345362e-01
  -3.40791699e-02 -2.29938173e-01]
 [-1.30667014e+00 -9.64789115e-01 -9.20764163e-01 ... -3.27632829e-01
  -2.97183209e-02 -2.23267001e-01]]
 modeling completed for speaker: shubham.gmm  with data point =  (1999, 40)
```

Figure 4.3: Gaussian Mixture Model with data point



```
Testing Audio :  shubham/shubham.wav
44100
[[ 0.0000000e+00  0.0000000e+00]
 [-3.0517578e-05  0.0000000e+00]
 [ 0.0000000e+00  0.0000000e+00]
 ...
 [ 6.7138672e-04  6.7138672e-04]
 [ 1.0070801e-03  1.0070801e-03]
 [ 1.3122559e-03  1.3122559e-03]]
[-20.41573233 -19.5524967  -15.38776268 -20.78433267 -18.88078128
 -10.07733636 -14.97994022 -15.44703809 -15.76218162 -19.23928135
 -17.46486303]
        detected as -  shubham
```

Figure 4.4: Log-likelihood of speakers

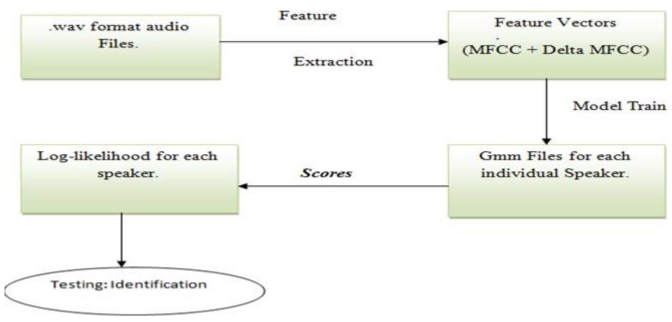


Figure 4.5: State transition Diagram

4.1.8 State Transition Diagram

4.1.9 Concepts

An example, There is a suspect on trial in court. He denies saying a word. The prosecution brings a record, saying that they've got their confession on tape. As the accused actively disputes his identity, an expert reveals exactly why no one else should have the identity. A scene made of fantasy! Maybe, but it's a factthat no two people in the world have exactly the same voices. Know why that'sso?

Our vocal cords can be in any of approximately 170 different positions. If the vocal cords are soft, they will vibrate about 80 times per second and the effect is deep tones. If tensed, they vibrate quickly, maybe 1000 times a second, creating short sound waves or high tones. The pitch of voice depends on the length of the vocal cord. Every voice has frequencies of a certain frequency. It is this range that defines a person's style of voice. The tongue 's movement against the palate, the shape of the lips and the alignment of the teeth often bring about changes in the voice.

This is the basic concept behind this big application of audio processing and voice recognition or identification / verification of speakers. Key features that differentiate two human voices can be extracted from a voice sample by extraction of the feature. This can also be used for model reference training; and lastly, the speaker recognition.

4.1.10 Data handling in Audio domain

As with all unstructured data formats, audio data has a few preprocessing steps to take before they are presented for review. We're going to get an idea here about why this is done.

The first step is to actually load the data into a comprehensible format for the computer. We simply take values for this after each particular step in the time.

For example; we extract values in a 2 second audio file in half a second. This is called the sampling of audio data, which is called the sampling rate at which it is sampled. Another way of expressing audio data is by translating it to another data representation domain, namely the frequency domain. When we sample an audio data, many more data points are needed to represent the entire data, and the sampling rate should be as high as possible.

On the other hand, if we represent audio data in the frequency domain it needs far less computational space. To get a preview of the photo below Here we divide

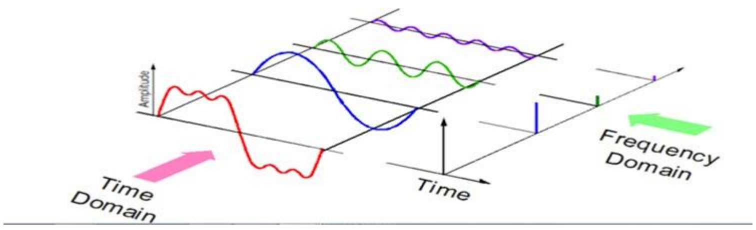


Figure 4.6: The unique value of frequency domain

one audio signal into three separate pure signals which can now be interpreted as three unique frequency domain values. There are a few more ways to represent audio data, for example using MFCs (Mel Frequency Cepstrum).

4.2 Implementation

4.2.1 Recording

```
import sounddevice as sd
from scipy.io.wavfile import write
fs = 44100 # Sample rate
print('recoding')
seconds = 10 # Duration of recording
myrecording = sd.rec(int(seconds * fs), samplerate=fs, channels=2)
sd.wait() # Wait until recording is finished
sd.stop()
print('Finished Recoding')
write('manish1.wav', fs, myrecording) # Save as WAV file
```

4.2.2 Features Extractions

```
import numpy as np
from sklearn import preprocessing
```

```
import python_ speech_ features as mfcc
def calculate_ delta(array):
"""Calculate and returns the delta of given feature vector matrix"""
rows,cols = array.shape
deltas = np.zeros((rows,20))
N = 2
for i in range(rows):
index = []
j = 1
while j <= N:
if i-j < 0:
first =0
else:
first = i-j
if i+j > rows-1:
second = rows-1
else:
second = i+j
index.append((second,first))
j+=1
deltas[i] = ( array[index[0][0]]-array[index[0][1]] + (2 * (array[index[1][0]]-array[index[1][1]]))
) / 10
return deltas
def extract_ features(audio,rate):
"""extract 20 dim mfcc features from an audio, performs CMS and combines delta to
make it 40 dim feature vector"""
mfcc_ feature = mfcc.mfcc(audio,rate, 0.025, 0.01,20,nfft = 1200, appendEnergy
= True)
print(mfcc_ feature)
mfcc_ feature = preprocessing.scale(mfcc_ feature)
delta = calculate_ delta(mfcc_ feature)
combined = np.hstack((mfcc_ feature,delta))
return combined
```

4.2.3 Training Model

```
import pickle
import os
import numpy as np
from scipy.io.wavfile import read
from sklearn.mixture import GaussianMixture
from featureextraction import extract_ features import
warnings
import sys
warnings.filterwarnings("ignore")
# path to training data
source = "voices"
# path where training speakers will be saved
dest = "trainmodel"
folders=os.listdir(source)
# print(folders)
train_ file = "developmentss.txt"
features_ file = "featuress.txt"
for fold in folders:
files=os.listdir(source+fold)
for file in files:
# print(file)
if os.path.exists(train_ file):
file_ paths.close()
file_ paths = open(train_ file,'r')
count = 0
features = np.asarray(())
features_ file_ features= open(features_ file,'a+')
# print(features)
for path in file_ paths:
path = path.strip()
print (path)
print(features)
# read the audio
sr,audio = read(source + path)
# extract 40 dimensional MFCC & delta MFCC features vector
= extract_ features(audio,sr)
if features.size == 0:
features = vector
else:
features = np.vstack((features, vector))
# when features of 5 files of speaker are concatenated, then do model training # if
count == -1:
print(features)
for s in features:
out_ arr = np.array_ str(s) features_ file_ features.write(out_ arr) #
sys.exit("Error message")
```

```
gmm = GaussianMixture(n_ components = 16, max_ iter = 200, covariance_
type='diag',n_ init = 3)
gmm.fit(features)
# dumping the trained gaussian model picklefile
= path.split("/")[0]+".gmm"
pickle.dump(gmm,open(dest+picklefile,'wb'))
print (' modeling completed for speaker:',picklefile," with data point = ",fea-
tures.shape)
features = np.asarray(())
count = 0
count = count + 1 file_
paths.close()
```

4.2.4 Test Recognition

```
import os
import pickle
import numpy as np
from scipy.io.wavfile import read
from featureextraction import extract_ features
import warnings
warnings.filterwarnings("ignore")
import time
source = "voices"
modelpath = "trainmodel" test_
file ="developmentss.txt" file_
paths = open(test_file,'r')
gmm_ files = [os.path.join(modelpath,fname) for fname in os.listdir(modelpath) if
fname.endswith('.gmm')]
models = [pickle.load(open(fname,'rb')) for fname in gmm_ files] speakers
= [fname.split("/")[-1].split(".gmm")[0] for fname in gmm_ files]
print("Do you want to Test a Single Audio: Press '1' or The complete Test Audio
Sample: Press '0' ?")
take = int(input().strip())
if take == 1:
print("Enter the File name from Test Audio Sample Collection :")
path = input().strip()
print ("Testing Audio : ", path)
sr,audio = read(source + path)
vector = extract_ features(audio,sr)
```

```
log_ likelihood = np.zeros(len(models))
for i in range(len(models)):
gmm = models[i] # checking with each model one by one
scores = np.array(gmm.score(vector))
log_ likelihood[i] = scores.sum()
winner = np.argmax(log_ likelihood)
print(log_ likelihood)
print ("detected as - ",speakers[winner])
time.sleep(1.0)
elif take ==0:
test_ file ="developmentss.txt"
file_ paths = open(test_ file,’r’)
for path in file_paths:
path = path.strip()
print ("Testing Audio : ", path)
sr,audio = read(source + path)
print(sr)
print(audio)
vector = extract_ features(audio,sr) log_
likelihood =np.zeros(len(models)) for i
in range(len(models)):
gmm = models[i] # checking with each model one by one
scores = np.array(gmm.score(vector))
log_ likelihood[i] = scores.sum()
winner = np.argmax(log_ likelihood)
print(log_ likelihood)
print ("detected as - ", speakers[winner])
checker_ name = path.split("_")[0]
print (" Speaker identified. ")
```

Chapter5

ResultsandDiscussions

5.1 DataDictionary

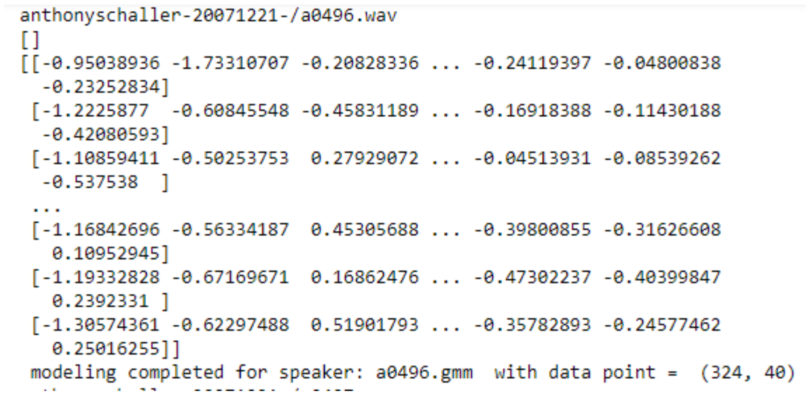
we have gathered two different datasets:

5.1.1 VoxForge

34 speakers each accompanied 5 voice samples for training data. And 5 voice samples for testing data. So, total of 340 cleaned and pre-processed voice sample Data. Each voice sample was around 4 sec. in length. So default value of nfft = 512 in mfcc() just worked fine.

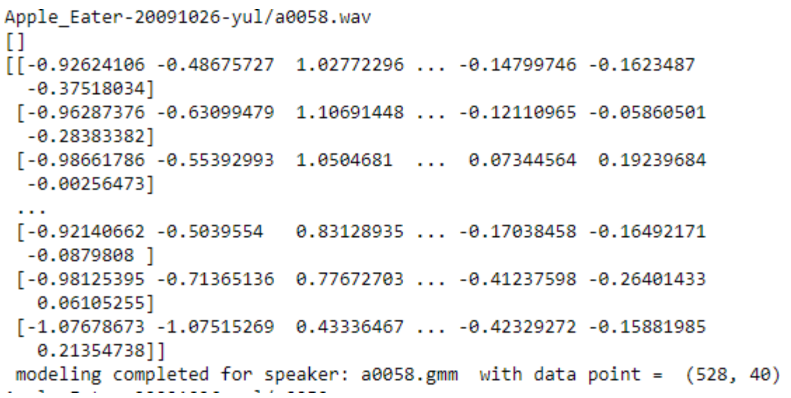
5.1.2 Self-made dataset

10 speakers each accompanied 5 voice samples for training data. And 5 voice samples for testing data. So, total of 50 voice sample Dataset. Each voice sample was around 30 sec. to 1 minute 40 sec in length. So, we had to tune parameters in mfcc(). Weadjusted nfft values from 512 to 2000 , finally.



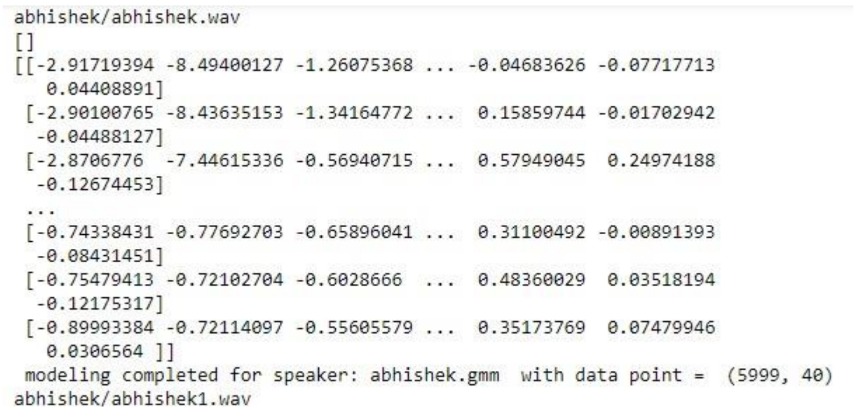
```
anthonyschaller-20071221-/a0496.wav
[]
[[-0.95038936 -1.73310707 -0.20828336 ... -0.24119397 -0.04800838
  -0.23252834]
 [-1.2225877  -0.60845548 -0.45831189 ... -0.16918388 -0.11430188
  -0.42080593]
 [-1.10859411 -0.50253753  0.27929072 ... -0.04513931 -0.08539262
  -0.537538  ]
 ...
 [-1.16842696 -0.56334187  0.45305688 ... -0.39800855 -0.31626608
   0.10952945]
 [-1.19332828 -0.67169671  0.16862476 ... -0.47302237 -0.40399847
   0.2392331 ]
 [-1.30574361 -0.62297488  0.51901793 ... -0.35782893 -0.24577462
   0.25016255]]
 modeling completed for speaker: a0496.gmm  with data point =  (324, 40)
```

Figure 5.1: VoxForge Modeling Diagram 1



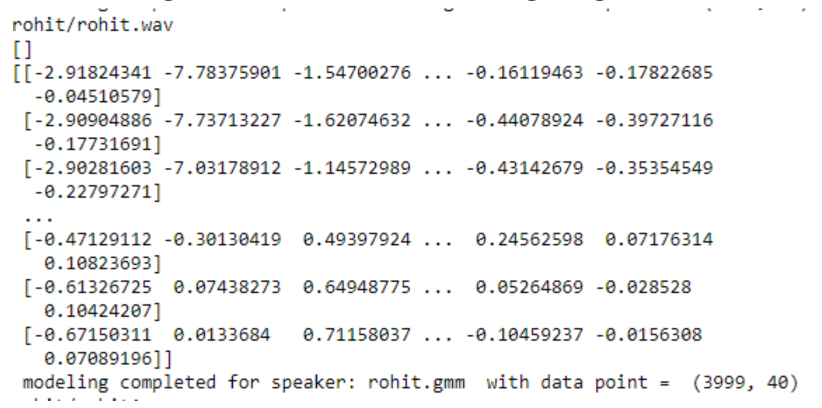
```
Apple_Eater-20091026-yul/a0058.wav
[]
[[-0.92624106 -0.48675727  1.02772296 ... -0.14799746 -0.1623487
  -0.37518034]
 [-0.96287376 -0.63099479  1.10691448 ... -0.12110965 -0.05860501
  -0.28383382]
 [-0.98661786 -0.55392993  1.0504681  ...  0.07344564  0.19239684
  -0.00256473]
 ...
 [-0.92140662 -0.5039554   0.83128935 ... -0.17038458 -0.16492171
  -0.0879808 ]
 [-0.98125395 -0.71365136  0.77672703 ... -0.41237598 -0.26401433
   0.06105255]
 [-1.07678673 -1.07515269  0.43336467 ... -0.42329272 -0.15881985
   0.21354738]]
 modeling completed for speaker: a0058.gmm  with data point =  (528, 40)
```

Figure 5.2: VoxForge Modeling Diagram 2



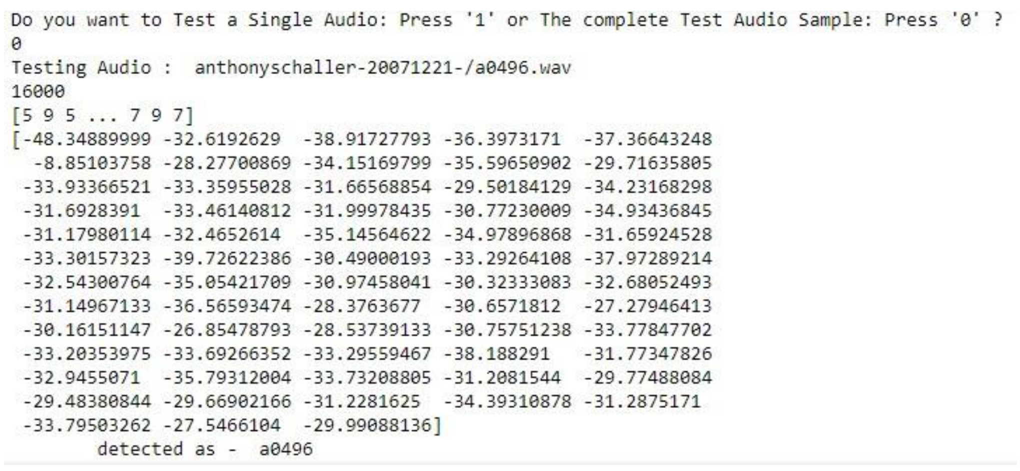
```
abhishek/abhishek.wav
[]
[[-2.91719394 -8.49400127 -1.26075368 ... -0.04683626 -0.07717713
   0.04408891]
 [-2.90100765 -8.43635153 -1.34164772 ...  0.15859744 -0.01702942
  -0.04488127]
 [-2.8706776  -7.44615336 -0.56940715 ...  0.57949045  0.24974188
  -0.12674453]
 ...
 [-0.74338431 -0.77692703 -0.65896041 ...  0.31100492 -0.00891393
  -0.08431451]
 [-0.75479413 -0.72102704 -0.6028666  ...  0.48360029  0.03518194
  -0.12175317]
 [-0.89993384 -0.72114097 -0.55605579 ...  0.35173769  0.07479946
   0.0306564 ]]
 modeling completed for speaker: abhishek.gmm  with data point =  (5999, 40)
abhishek/abhishek1.wav
```

Figure 5.3: Self-made Modeling Diagram 1



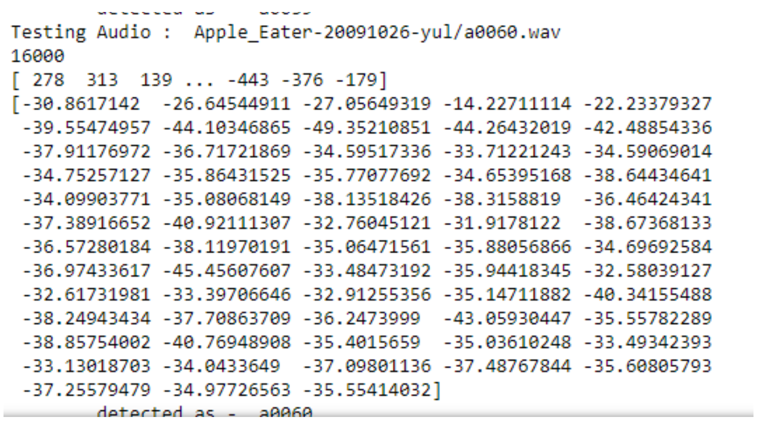
```
rohit/rohit.wav
[]
[[-2.91824341 -7.78375901 -1.54700276 ... -0.16119463 -0.17822685
  -0.04510579]
 [-2.90904886 -7.73713227 -1.62074632 ... -0.44078924 -0.39727116
  -0.17731691]
 [-2.90281603 -7.03178912 -1.14572989 ... -0.43142679 -0.35354549
  -0.22797271]
 ...
 [-0.47129112 -0.30130419  0.49397924 ...  0.24562598  0.07176314
   0.10823693]
 [-0.61326725  0.07438273  0.64948775 ...  0.05264869 -0.028528
   0.10424207]
 [-0.67150311  0.0133684   0.71158037 ... -0.10459237 -0.0156308
   0.07089196]]
 modeling completed for speaker: rohit.gmm  with data point =  (3999, 40)
```

Figure 5.4: Self-made Modelng Diagram 2



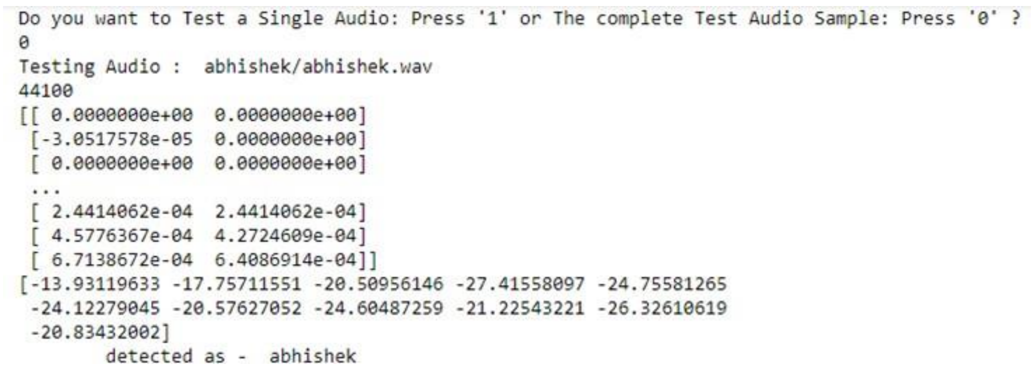
```
Do you want to Test a Single Audio: Press '1' or The complete Test Audio Sample: Press '0' ?
0
Testing Audio :  anthonyschaller-20071221-/a0496.wav
16000
[5 9 5 ... 7 9 7]
[-48.34889999 -32.6192629  -38.91727793 -36.3973171  -37.36643248
  -8.85103758 -28.27700869 -34.15169799 -35.59650902 -29.71635805
 -33.93366521 -33.35955028 -31.66568854 -29.50184129 -34.23168298
 -31.6928391  -33.46140812 -31.99978435 -30.77230009 -34.93436845
 -31.17980114 -32.4652614  -35.14564622 -34.97896868 -31.65924528
 -33.30157323 -39.72622386 -30.49000193 -33.29264108 -37.97289214
 -32.54300764 -35.05421709 -30.97458041 -30.32333083 -32.68052493
 -31.14967133 -36.56593474 -28.3763677  -30.6571812  -27.27946413
 -30.16151147 -26.85478793 -28.53739133 -30.75751238 -33.77847702
 -33.20353975 -33.69266352 -33.29559467 -38.188291   -31.77347826
 -32.9455071  -35.79312004 -33.73208805 -31.2081544  -29.77488084
 -29.48380844 -29.66902166 -31.2281625  -34.39310878 -31.2875171
 -33.79503262 -27.5466104  -29.99088136]
        detected as -  a0496
```

Figure 5.5: VoxForge detection Diagram 1



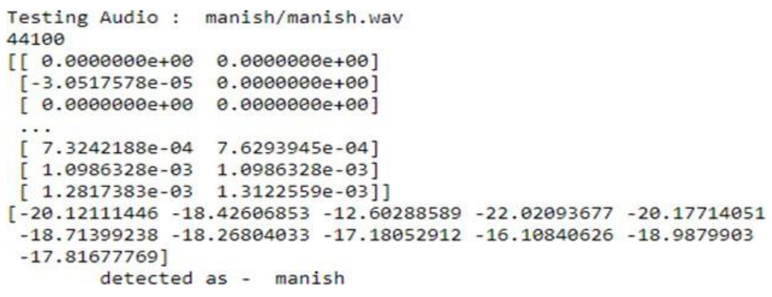
```
Testing Audio :  Apple_Eater-20091026-yul/a0060.wav
16000
[ 278  313  139 ... -443 -376 -179]
[-30.8617142  -26.64544911 -27.05649319 -14.22711114 -22.23379327
 -39.55474957 -44.10346865 -49.35210851 -44.26432019 -42.48854336
 -37.91176972 -36.71721869 -34.59517336 -33.71221243 -34.59069014
 -34.75257127 -35.86431525 -35.77077692 -34.65395168 -38.64434641
 -34.09903771 -35.08068149 -38.13518426 -38.3158819  -36.46424341
 -37.38916652 -40.92111307 -32.76045121 -31.9178122  -38.67368133
 -36.57280184 -38.11970191 -35.06471561 -35.88056866 -34.69692584
 -36.97433617 -45.45607607 -33.48473192 -35.94418345 -32.58039127
 -32.61731981 -33.39706646 -32.91255356 -35.14711882 -40.34155488
 -38.24943434 -37.70863709 -36.2473999  -43.05930447 -35.55782289
 -38.85754002 -40.76948908 -35.4015659  -35.03610248 -33.49342393
 -33.13018703 -34.0433649  -37.09801136 -37.48767844 -35.60805793
 -37.25579479 -34.97726563 -35.55414032]
        detected as -   a0060
```

Figure 5.6: VoxForge detection Diagram 2



```
Do you want to Test a Single Audio: Press '1' or The complete Test Audio Sample: Press '0' ?
0
Testing Audio :  abhishek/abhishek.wav
44100
[[ 0.0000000e+00  0.0000000e+00]
 [-3.0517578e-05  0.0000000e+00]
 [ 0.0000000e+00  0.0000000e+00]
 ...
 [ 2.4414062e-04  2.4414062e-04]
 [ 4.5776367e-04  4.2724609e-04]
 [ 6.7138672e-04  6.4086914e-04]]
[-13.93119633 -17.75711551 -20.50956146 -27.41558097 -24.75581265
 -24.12279045 -20.57627052 -24.60487259 -21.22543221 -26.32610619
 -20.83432002]
        detected as -   abhishek
```

Figure 5.7: Self-made detection Diagram 1



```
Testing Audio :  manish/manish.wav
44100
[[ 0.0000000e+00  0.0000000e+00]
 [-3.0517578e-05  0.0000000e+00]
 [ 0.0000000e+00  0.0000000e+00]
 ...
 [ 7.3242188e-04  7.6293945e-04]
 [ 1.0986328e-03  1.0986328e-03]
 [ 1.2817383e-03  1.3122559e-03]]
[-20.12111446 -18.42606853 -12.60288589 -22.02093677 -20.17714051
 -18.71399238 -18.26804033 -17.18052912 -16.10840626 -18.9879903
 -17.81677769]
        detected as -   manish
```

Figure 5.8: self-made detection Diagram 2

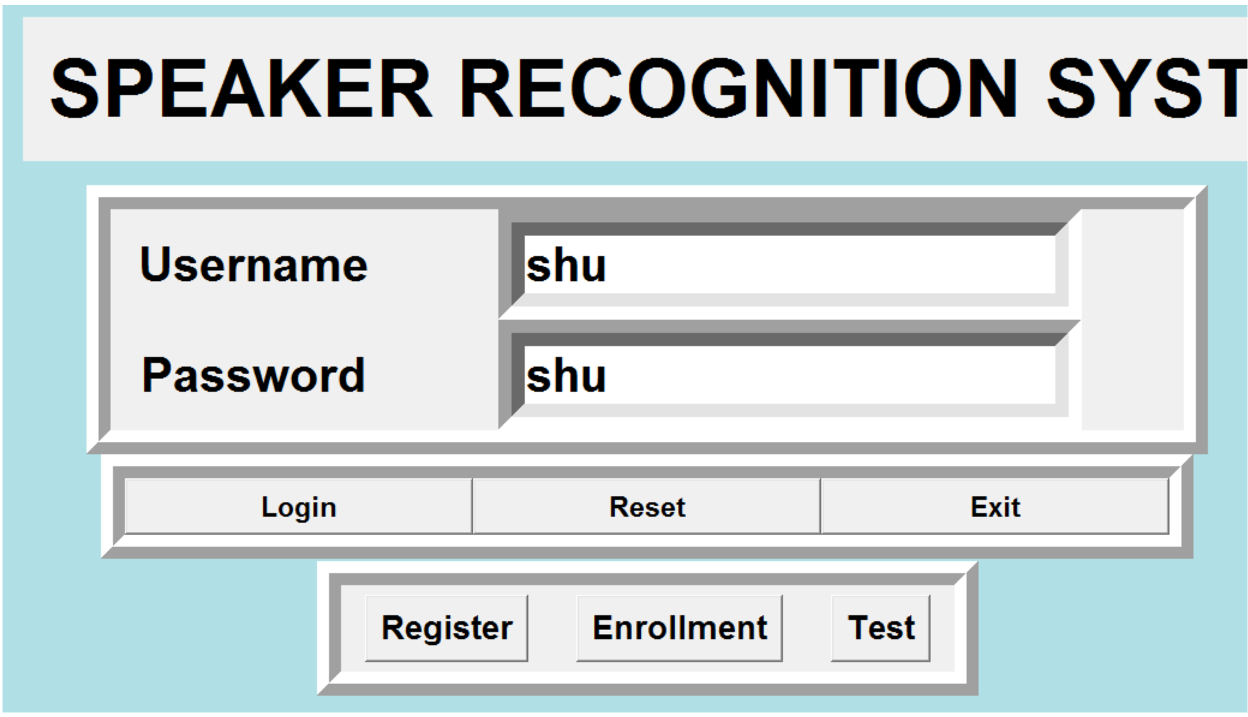


Figure 5.9: self-made GUI speaker Recognition System

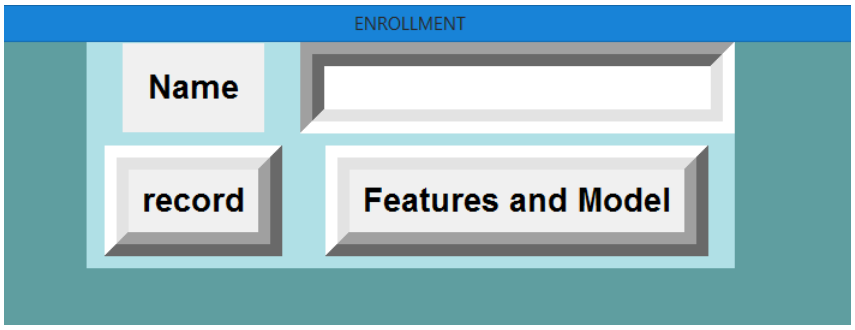


Figure 5.10: self-made GUI speaker Recognition System II

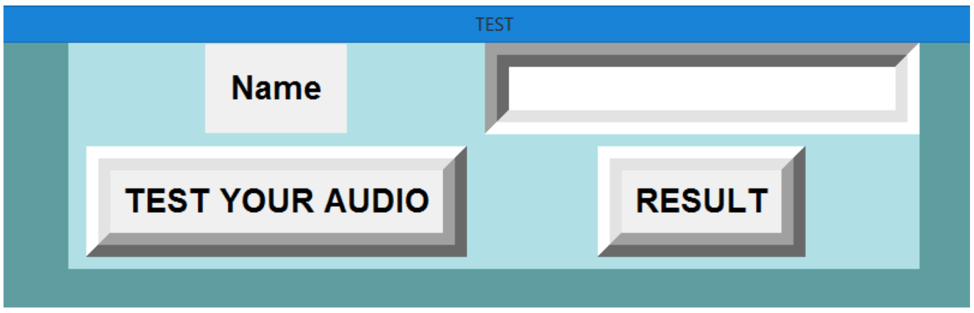


Figure 5.11: self-made GUI speaker Recognition System III

Chapter6

ConclusionandFutureWork

6.1 Conclusion

We also developed a method for developing a reliable and more accurate speaker recognition program. The suggested structure sets out a system for identifica- tion of speakers. Speaker registration is the first step in developing a speaker recognition program, in which each applicant registered receives a collection of statements. For training, an equivalent number of speaking templates is generated for each speaker (for individual speaking templates it does not need to be of the same duration) but it may differ for testing. Speaker's series of models is used as an individual pattern.

System selects the template whose match score is more closed to the test template during matching process. The proposed structure outlines the overall process involved in creating a program for voice recognition. The proposed system has long-term methodological credibility. In the context of speaker recognition, either static or dynamic characteristics of speech features are used. The proposed structure has the potential for more embodiment, so it would be possible to change as needed.

The speaker identification device output using different techniques for extracting and matching features. Our program uses MFCC algorithm because it has the least false acceptability ratio. I n can be used in function matching technique in order to boost device efficiency and also to achieve high precision GMM model. The speakers were equipped and tested using model MFCC and model GMM. They offer a better rate of identification for features of speakers. This technique combined the pitch information with MFCC in future research, and also evaluated the output of the speaker identification device in the presence of noise. This paper gives an overview of the speaker recognition that includes various methods of features extraction and model training mainly focused on MFCC and GMM.

Security puts great setbacks for sensitive information in today's world. Recognition of speakers is a multidisciplinary biometrics branch that can be used to clas- sify and validate speakers to protect sensitive information. Therefore to prevent unauthorized access, a voice-based recognition system needs to be developed that provides a solution for financial transaction and protection of personal data that would minimize theft. MFCC features extraction technique and GMM modeling technique use in speaker recognition are discussed in this paper which can be applied to develop a real time. application for speaker identification and verification system for confidential data securing in the future.

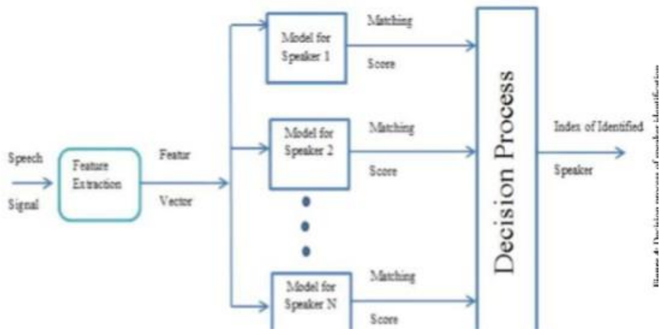


Figure 6.1: Decision process of speaker recognition

6.2 Future scope

The area of Speaker Recognition is really fascinating and to explore. But, as per the current scenario, we have had very limited research in this area. In this sector, therefore, there is a lot to be done. We are dealing with a big chunk of audio data every day. Plus, we could never leave behind synthesizing speeches and honoring speakers in this progressing country.

For ease in user experience, my project can be expanded further with a GUI interface. If we think about its applicability it will have tremendous potential for the future. The method followed, and my project results would be useful to other researchers. They will consider and suggest new ways of improving model accuracy and consistency.

This specific project can be used as a biometric voice device for detecting and catching offenders. I have a vision of further expanding this initiative by adding Speech Recognition to the initiative. Speech Recognition + Spokesperson Recognition will suit my purpose perfectly. I 'd like to make an intelligent device that would start tapping phone calls while receiving some weak keywords, oftenused by terrorists; then that terrorist or other criminal will be recognized by the speaker recognition program. This will aid our country immensely in rising crime and terror attacks.

The results indicate fairly strong awareness Continuous speech with a broad vocabulary, by various speakers. The Specific modules were analyzed and analysed in their respective domains Checked successfully for the various speech input files. We figured out the 4 stages Recognition of speech-general diagnosis, preprocessing, phoneme Computer Recognition and Text Recognition. Independent Voiced recognition systems were equipped to identify speech effectively Inputs reported using a microphone and spoken specimens Retrieved from repositories. Key challenges for future research include acoustic robustness, use Multiple word pronunciations and successful access restrictions Extremely broad lexicon and well developed methods for conceptual extraction Word-hypothesized representations. The reconnaissance device seen in our proposed method conducts independent speech by speaker stage mode Reconnaissance. The results obtained can be improved with fine tuning of the program For wider repositories for the instruction. The next move will be live acknowledgement Redeeming this willtake more resources like greater expression Databases, acoustic models and extensive vocabulary to provide strong Performance onacceptance.

AppendixA

Appendices

A.1 Why wreck a nice beach? Why recognize speech or speaker? Some important concepts for the speech recognition module

Youcan not help but understand your mother tongue. If anyone talks to you in your mother tongue under normal circumstances, it's almost difficult to hear the voice as a meaningless series of sounds — the spoken communication — you hear words and you understand. Nearly anyone with normal hearing will understand the spoken language; it's one of your day's most common and important tasks. And how easily you get used to a new accent, and even start taking it on, is in- credible.
Contrast the ease of knowing your mother tongue with the constant sound stream that you hear while listening to a language you don't know. You can not tell where the words start and end. Youcan choose a sentence boundary once in a while be- cause of the intonation or a pause. Computers attempting to understand speech are facing a steady stream of sound that they must break into words, without ben- efiting from a language awareness of the brain or nativespeaker.

A.1.1 Whatspelling doesn't represent

Much like a road map that abstracts from geographical fact, abstracts written language from spoken language. Until looking at the next sentence, think for a moment about the kinds of things that are not reflected in writtenEnglish.
Below are some tips if you haven't thought much about it: Which kinds of directions for the stage and the actor are included in a play script? How many different ways may anyone say "You 're a fine one to talk." It can be said simply, sarcas-

tically, joyfully, violently, skeptically, or fearfully, none of which is reflected in our system of spellings. Spelling does not represent feeling, speaking pace, or personality, nor does it represent intonation or voice tone.

A.1.2 What spelling doesn't represent consistently

Bad spellers know English is quite an imperfect system. Spelling, as many of you know, does not always operate with a one-to-one connection between soundsand letters. This old poem exemplifies some of the English spellingdifficulties.
Words with the same spelling can have radically different pronunciations, such as: cough, heavy, bough, though, thoroughfare (Finegan 1992). Phonetic transcription for these kinds of words lets you represent the pronunciation unambiguously. Spelling is a problem for English-speaking grade schoolers — it must be taught, and often with great difficulty. The spelling bee does not occur in countries such as Spain and Indonesia, whose national languages have a highly clear link between sound and spelling.

A.1.3 The basics of speech production and perception

When we force air out of the lungs through the glottis (vocal cords/larynx), it causes the vocal cords to vibrate; this vibration in turn creates air pulses that escape through the mouth (and often through the nose); these pulses are simply slight fluctuations in the air pressure due to the air molecules' wavelike motion. "Wecan explain the sounds we produce as to how easily changes in the vibration of the air happen, which defines the fundamental sound frequency and is heard by the heart as a pitch. Wecan also define the magnitude or strength of the changes that influence sound loudness.
The sound quality depends on the shape of the vocal tube as the air passes through. Speaking spectrographs are an significant tool in acoustic science. A "fotos" is made of the speech signal when you speak to a microphone connected to the unit. The created patterns are called spectrogramming or "visible expression" more vividly.

Bibliography

[1]George R Doddington, Member, IEEE, “Speaker Recognition- Identifying People by their Voices”, proceedings of the IEEE, Vol. 73, no. 11, pp. 1651-1664, November 1985.

[2]Richard D. Peacocke, and Daryl H. Graf, “An Introduction to Speech and Speaker Recognition “, IEEE, pp. 26-33, August 1990.

[3]Tomi kinnunen, Haizhou Li, “An overview of textindependent speaker recognition: From features to supervectors”, Speech Communication 52, pp. 12-40, 2010.

[4]Ling Feng , Kgs. Lyngby “Speaker Recognition”, Thesis, Technical University of Denmark Informatics and Mathematical Modeling, Denmark,2004

[5]J. Wu and J. Yu, "An improved arithmetic of MFCC in speech recognition system," in Electronics, Communications and Control (ICECC), 2011 International Conference on, 2011, pp. 719-722

[6]Yingjie He, Liwei Ding, Yuxian Gong, Yongjin Wang,“Real-time Audio & Video Transmission System Based on Visible Light Communication ”,June 2013.

[7]Dalei Wu, Andrew Morris, Jacques Koreman; "MLP Internal Representation as Discriminative Features for Improved Speaker Recognition",2005.

[8]Mitchell McLaren; Advances in deep neural network approaches to speaker recognition, ICASSP– April, 2015.

[9] David Moffat, David Ronan, Joshua D. Reiss ;AN EVALUATION OF AUDIO FEATURE EXTRACTION TOOLBOXES; Nov 30 - Dec 3, 2015

[10] B.Mathieu, S.Essid, T.Fillon, J.Prado, G.Richard; YAAFE, an Easy to Use and Efficient Audio Feature Extraction Software, proceedings of the 11th ISMIR conference, Utrecht, Netherlands, 2010.