論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Comprehensive Feature Optimization for Robust Speaker Recognition Across Diverse Datasets
著者(和文)	CHAUHANNeha
Author(English)	Neha Chauhan
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第12859号, 授与年月日:2024年9月20日, 学位の種別:課程博士, 審査員:一色 剛,髙橋 篤司,本村 真人,原 祐子,佐々木 広
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第12859号, Conferred date:2024/9/20, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Doctoral Dissertation

Comprehensive Feature Optimization for Robust Speaker Recognition Across Diverse Datasets

Neha Chauhan

Information and Communications Engineering, Tokyo Institute of Technology

Supervisor: Professor Tsuyoshi Isshiki

February 2024

Abstract

This thesis presents a comprehensive exploration of advanced techniques to enhance the performance of speaker recognition models. The research is divided into two major parts, each emphasizing distinct methodologies to improve the accuracy and efficiency of speaker recognition systems and other part is reducing computational timing.

In the initial part, a novel speaker recognition model is proposed, leveraging the fusion of various speech features. A unique feature aggregation method, encompassing 18 features such as mel frequency cepstral coefficient (MFCC), linear predictive coding (LPC), perceptual linear prediction (PLP), root mean square (RMS), centroid, and entropy features, along with their delta (Δ) and delta–delta (ΔΔ) feature vectors, is introduced. The efficacy of this approach is evaluated across five speech datasets—NIST-2008, voxforge, ELSDSR, VCTK, and voxceleb1—using MATLAB classification learner application with linear discriminant (LD), K nearest neighbor (KNN), and ensemble classifiers. Notably, the LD classifier achieves a speaker identification (SI) accuracy of 96.9% and 100% for NIST-2008 and voxforge datasets, respectively, along with the lowest speaker verification (SV) equal error rate (EER) values. The fusion of diverse features demonstrates a significant increase in speaker identification accuracy (10–50%) and a reduction in SV EER compared to single-feature approaches.

The second part of the thesis provides a comprehensive examination of speaker recognition, emphasizing three approaches: feature-level fusion, dimension reduction using principal component analysis (PCA) and independent component analysis (ICA), and feature optimization employing genetic algorithm (GA) and marine predator algorithm (MPA). Evaluation is conducted on diverse speech datasets with varying noise levels and speaker counts. The results underscore the effectiveness of incorporating PCA with GA and MPA, showcasing notable improvements in speaker recognition performance. Optimal outcomes are achieved across different datasets and classifiers, with features such as TIMIT babble noise (120 speakers) attaining a speaker identification accuracy of 92.7% using feature fusion and a speaker verification EER of 0.7% through various feature optimization techniques with LD and KNN classifiers.

This study not only enhances speaker recognition accuracy but also significantly improves computational efficiency, particularly for large-scale datasets like voxceleb1. The insights gained from this research contribute valuable knowledge for the development of effective speaker recognition systems.

Acknowledgments

I extend my heartfelt gratitude to Professor Tsuyoshi Isshiki, my esteemed thesis supervisor, for his unwavering support and invaluable guidance during the challenging phases of my research journey.

I am deeply grateful to Professor Dongju Li for their insightful feedback and exceptional suggestions, which significantly enriched the quality of this work.

I am indebted to the members of my laboratory at the Tokyo Institute of Technology for their stimulating discussions, relentless assistance, and collaborative spirit, all of which have contributed to the success of this research endeavor.

I would like to express my profound appreciation to my family and friends for their unending encouragement and unwavering support, which have been the cornerstone of my perseverance during difficult times.

I am also grateful to the broader academic community for their contributions and inspiration that have shaped my academic pursuits and aspirations.

Content

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Speech processing is the comprehensive study of various types of speech signals and their corresponding processing methods. Typically, these signals undergo digital representation, making speech processing a specialized branch of digital signal processing. One key application of speech processing is speaker recognition, which involves automatically identifying speaker identity based on recorded voice samples. This technology enables access control for services like voice dialing, database access, information services, voice mail, security control, remote computer access, and other domains prioritizing security [1].

Speech is a complex signal resulting from multiple transformations occurring at different levels, including semantic, linguistic, articulatory, and acoustic aspects. Variations in these transformations manifest as differences in the acoustic properties of the speech signal. Moreover, speaker-specific differences arise from a combination of anatomical variations within the vocal tract and individual speaking habits. Speaker recognition accounts for these differences to distinguish between speakers. [2]. The subsequent chapters delineate the construction of a simple yet effective automatic speaker recognition (ASR) system. Such a system contributes significantly to speaker identification, an essential aspect of the speaker recognition field. The system's potential applications extend to numerous security-related areas. Speech serves as the primary medium for human communication, with a complex structure encompassing not only voice transmission but also gestures, language, topics, and listener comprehension capabilities. Consequently, researchers over the past five decades have delved into various facets of speech, including the mechanical realization of speech signals, human-machine interaction, and speech and speaker recognition.

As computers and processor-based personal devices, such as cell phones, continue to proliferate, the scientific community strives to develop human-like interfaces. This endeavor demands precise speech recognition, ideally language and speaker-independent, for command and query purposes, speaker recognition for speaker separation and authentication, and text-to-speech conversion with human-like quality. One of the primary objectives of speaker recognition is to enable communication with machines via voice. Despite existing means of machine communication, such as keyboards and other devices, being relatively slow, speech plays a pivotal role in streamlining this process [1], [2]. It can be categorized further into speaker identification versus speaker verification, text-dependent versus text-independent, and closed-set versus open-set speaker recognition systems (figure 1) [1],[2].

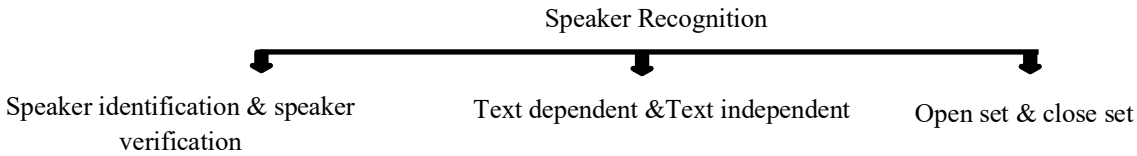


Fig.1 Classification of speaker recognition system.

Speaker recognition systems can be categorized based on various factors including the nature of speech, the context in which the system operates, and the extent of flexibility it offers in recognizing speakers. Here is an enhanced version of the categorization.

1.1.1 Speaker identification vs. Speaker verification

Speaker identification refers to the process of determining the identity of a speaker from a set of known speakers. It involves comparing a given speech sample with a database of known speakers to determine the most likely match. This process is often used in forensic investigations, security systems, and other applications where determining the identity of a speaker is critical. Speaker identification can be achieved using various techniques, including pattern recognition, machine learning algorithms, and voice biometrics. While Speaker verification, on the other hand, involves confirming or verifying the claimed identity of a speaker. In this process, the system compares the speech sample provided by the speaker with a pre-recorded sample of the same speaker to authenticate their identity. Speaker verification is commonly used in security systems, access control, and

other applications where the verification of a speaker's identity is essential. Various algorithms and methods, such as text-dependent and text-independent systems, are used for speaker verification. Figure 2 and figure 3 shows how speaker identification and speaker verification system works respectively.

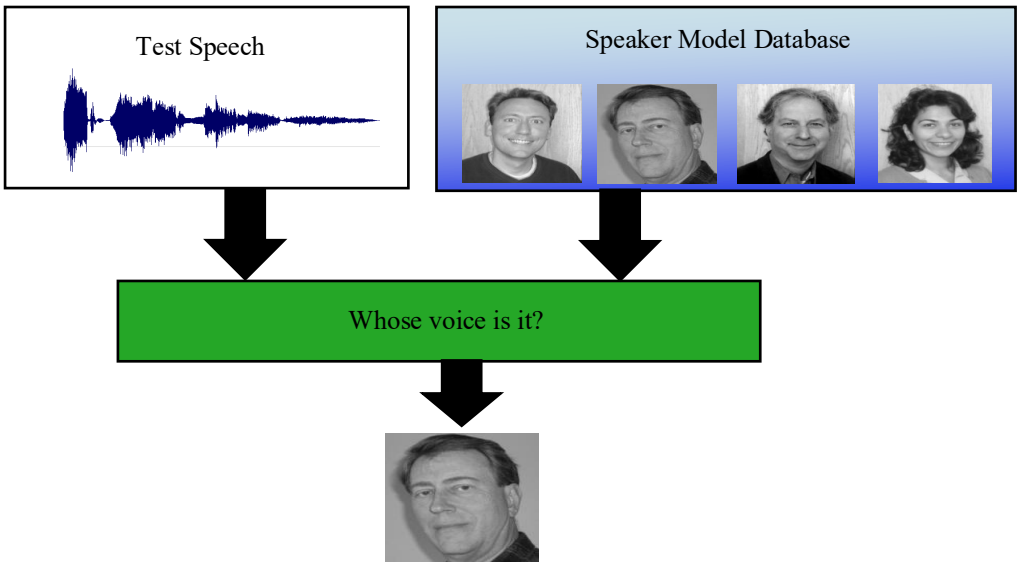


Fig.2 Speaker Identification.

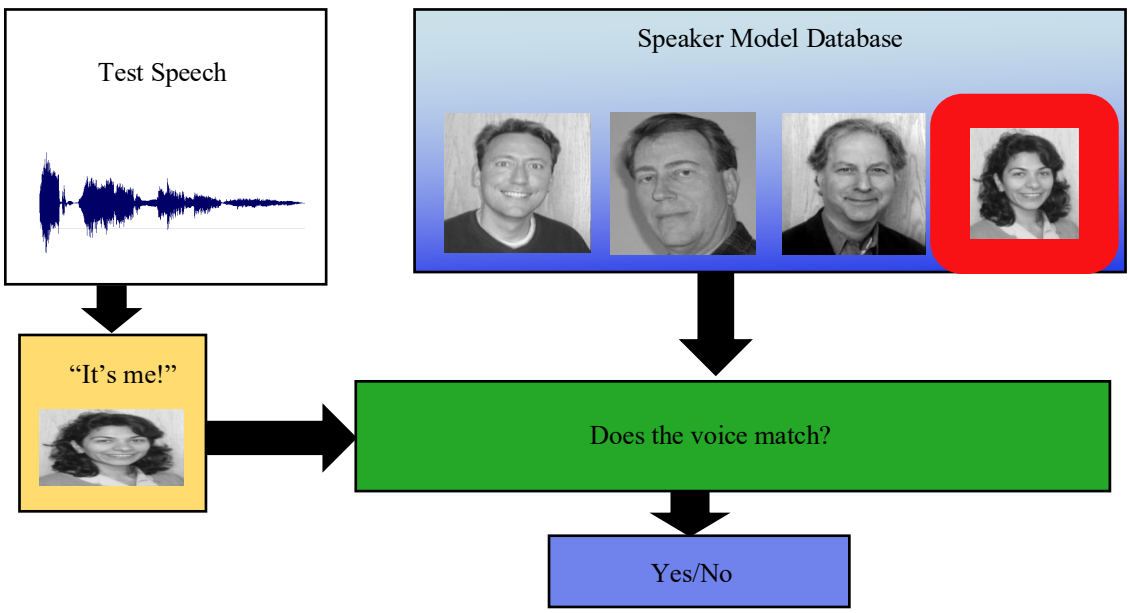


Fig.3 Speaker Verification.

1.1.2 Text-Dependent vs. Text-Independent

Text-Dependent systems rely on specific predefined phrases or content for recognition. The speaker must utter specific passphrases or words during the identification process. These systems are commonly used in applications such as voice-access ATMs for heightened security.

Text-Independent systems, on the other hand, do not depend on any specific content within the speech. The speaker can freely articulate any words or sentences during the identification process. These systems are more suitable for applications where natural speech needs to be recognized, such as in forensic investigations or surveillance.

1.1.3 Open Set vs. Closed Set

Open Set systems can handle an indefinite number of registered speakers. Authorized individuals might not necessarily be part of the initially registered group of speakers. These systems are useful in scenarios where the speaker database may expand over time, such as in large-scale voice recognition systems for public services or multi-user applications.

Closed Set systems, however, have a fixed and limited number of pre-registered users. The system only recognizes speakers who are already enrolled in the system. These systems are typically utilized in controlled environments with a predetermined group of users, like secure facilities or restricted-access systems. Each type of speaker recognition system caters to specific requirements and contexts, allowing for flexibility and adaptability in various applications while ensuring the desired level of security and usability [2],[3].

1.2 Related work

1.2.1 Advancements in integrating speech features, dimensionality reduction, and feature optimization for enhanced speaker recognition systems.

In recent years, speaker recognition systems have gained significant prominence in a variety of applications, ranging from security measures to forensic analysis [4]. The ability to accurately distinguish and verify individuals based on their unique vocal characteristics has become a focal point of extensive research and technological development. This research emphasized the integration of multiple speech features, including mel frequency cepstral coefficient (MFCC), linear predictive coding (LPC), perceptual linear prediction (PLP), root mean square (RMS), centroid, and entropy features, to enhance the precision of speaker recognition systems, thus resulting in significant improvements in identification rates [4].

One key challenge encountered in the existing research was the increased computation time required for data classification, particularly as the dimensionality of the features increased. To overcome this challenge, dimensionality reduction techniques such as principal component analysis (PCA) and independent component analysis (ICA) and features optimization techniques are employed. These techniques compress the data by reducing its dimensionality while preserving the most important features. Consequently, the training process is accelerated, leading to improved computational efficiency [5]. The proposed research aims to address the limitations of previous studies, which predominantly utilized clean data, by conducting experiments with both noisy data and the widely-used voxceleb dataset. This approach ensures a fair comparison of results with the existing research papers.

By combining dimensionality reduction and feature optimization techniques, thesis addresses the challenges of computational complexity and improves the overall speed and effectiveness of the speaker recognition system. This approach enables faster classification of data, making real-time or near real-time applications feasible. Furthermore, the paper introduces feature optimization methods such as genetic algorithms (GA) and marine predator algorithm (MPA) for selecting the best features for the speaker recognition system with dimension reduction techniques. These algorithms help identify the most informative and discriminative features, enhancing the overall performance of the system. Optimizing the feature selection process improves the accuracy and efficiency of speaker identification and verification [6],[7],[8], [9]. Additionally, the thesis explores the use of dynamic features, including delta and delta-delta features, to capture temporal variability in speech signals and improve speaker recognition performance.

1.2.2 Common methods for extracting features in speaker recognition

Automatic speaker recognition (SR) is a state-of-the-art technique [10, 11]. Feature extraction and feature mapping are two important processes in SR. Feature extraction extracts several feature vectors called descriptors, and feature matching is used to avoid redundancy present in speech signal features and is used to compare the feature vectors extracted from a signal belonging to an unknown speaker with those extracted from a signal belonging to a known speaker set [12-14]. In the 1980s, the mel frequency cepstral coefficient (MFCC) was introduced; they use the mel frequency scale, which is a characteristic of popular speech [15]. A comparison of various features, such as the MFCC, the linear prediction cepstral coefficient (LPCC), linear predictive coding (LPC) and the perceptual LPCC (PLPCC), for SR was made. It was concluded that among all these features, MFCC and LPCC allowed for better performance than other features [15]. The concept of dynamic features was introduced in 1981 by Furui [16] to detect the temporal variability in feature vectors. In addition, in short-term frame energy,

the formed transitions and energy modulations also include useful speaker information [16]. The major problem is the deterioration of the performance of ASR systems in the presence of additive noise [17]. Overall, researchers have tried to ensure that recognition systems are noise-resistant in three main ways: (a) statistical models have been adapted to recognize noise (e.g., by using parallel model combinations) [18], (b) methods for decreasing the noise in speech signals have been proposed [19, 20], and (c) noise-resistant features have been applied. Several techniques have been designed to address the sensitivity of cepstral features to noise, and various approaches, such as wiener filtering [21], spectral subtraction [22], RASTA [23], and lin-log RASTA [24], have been proposed. The improvement in the cepstral features themselves has not produced satisfactory results. Because of this limitation, more research has been carried out on new features that are more robust to noise in addition to cepstral features.

MFCC, LPC and PLP features extract most of the important information from speech signals and cover information of the vocal tract. However, these features do not provide information about the vocal system. To build a better SR model, it is important to extract additional speaker-dependent information, such as entropy, energy, centroid and prosodic information, with the root mean square (RMS) values. Prosodic features, such as pitch, energy, RMS and duration, are comparatively less disturbed by channel differences and noise. Although systems based on spectral features, such as MFCC, perform better than prosody-based systems, their combined performance may provide the robustness needed by recognition systems [25-26]. Prosodic features are those features of speech that deal with the auditory properties of sound, such as stress and pitch, which are different features [31].

One of the important problems related to the degradation of the performance of SR is that the MFCC, LPC, PLP, centroid, entropy, and RMS feature set contains only static features. A static feature does not capture small changes in the speech signals because the speech signals change very frequently; consequently, the values of the signals also change rapidly. Therefore, delta and delta-delta feature values are used to add more information and detect feature values over small intervals in speech signals [27]. Many research papers show that models with delta values drastically improve the performance of SI/SV systems [27, 31]. However, from the results of many published research papers [29, 33, 34] and their references, it is clear that prosodic information can also be used to improve SR performance. Feature-level fusion and score-level fusion are two important fusion levels in a biometric system [35]. Usually, feature-level fusion contains more information than a single feature of the speaker's voice and thus improves the performance of the SR system [36]. This concept of information theory for SR systems is explained clearly in [37, 38]. The main goal of this research is to enhance the speaker identification (SI) accuracy and reduce the equal error rate (EER) value because various experiments have been performed using feature-level fusion.

1.2.3 Approaches utilized in speaker recognition system with ELSDSR, VCTK, voxforge, NIST-2008, voxceleb1, and TIMIT Speech Databases

Here, we give a detailed overview of the popular methods used for SI and SV systems, mainly using the ELSDSR, VCTK, voxforge, NIST-2008, voxceleb1and TIMIT speech databases. The idea behind the feature-level combination scheme is that each feature contains some aspect of speaker information that could be missed by others. Within this scheme, several speaker-specific features are concatenated to construct models and for comparison. Furui first used the feature concatenation approach for the joint use of cepstral and polynomial features in the form of delta and delta-delta coefficients [16]. The concatenation of MFCC and spectral features enhances the performance of the SR system [39]. In [40], it is shown that the concatenation of phase information with MFCC enhances speaker performance [40]. In another work, the authors jointly used the statistical pH feature and the concatenation characteristics of MFCC and achieved better performance under noise conditions [41].

In [42], SI implementation of score-level fusion and feature-level fusion is performed with ELSDSR audio data. The score-level fusion-based system gave a better identification rate of 100% than all other systems using a support vector machine (SVM) [42]. Score-level fusion and feature-level fusion were used in [43] to calculate SI accuracy using ELSDSR speech data, and the SI accuracy was increased to 98% and EER reduces to 2% when score-level fusion was used in random forest (RF) and multiclass SVM classification.

In [44], the authors show the potential of deep belief networks (DBNs) in the extraction of short-term spectral features. An accuracy of 95% is achieved by combining MFCC and DBN features with the gaussian mixture model-universal background model (GMM-UBM) on the ELSDSR database. In [45], a two-step approach using the gender and voice information of speakers from ELSDSR was proposed, and the model obtained an improved accuracy of 99.9% with the GMM classifier. Paper [46] presented a simulation study on a transformation-based fusion algorithm for a multimodal biometric authentication system using an ensemble classifier with face scores and voice recognition modules. Using score fusion, true positive rates of 99% and an accuracy of 99.22% are achieved on the ELSDSR voice dataset. [47] used the score fusion method with SVM, linear discriminant analysis (LDA) classifier and MFCC, delta MFCC, and delta-delta MFCC features for GMM-UBM modeling. The best EER of 0.02 is obtained with LDA and cosine distance scoring. In [48], a new type of pipeline architecture

was proposed, and the fusion of the gabor filter (GF) and convolutional neural network (CNN) features with RF, SVM, and deep neural network (DNN) classifiers on the ELSDSR database is used. The best accuracy of 94.87% is obtained using the RF classifier for 22 speakers.

In [49], the authors proposed a new type of feature extraction technique called the twofold information set (TFIS) for a text-independent SR system on three voice datasets, i.e., NIST-2003, voxforge (2015), and VCTK. On the voxforge 2014 database, for the clean voice dataset, the best performance accuracy of 100% and an EER of 0.02 were achieved. On the VCTK dataset, the best SI accuracy of 98.9% and an EER of 0.05 were achieved using TFIS features, and a genuine acceptance rate (GAR) of 0.1% was achieved. The closed-set text-independent SI system (CISI) based on a multiple classifier system (MCS) was proposed in [50] using the expectation-maximization (EM) algorithm. As a result, the best SI accuracy of 97% was achieved with a hybrid modeling method of vector quantization (VQ) and a gaussian mixture model (GMM).

Reference [51] showed how the accuracy of SI improves when the fusion of delta and delta-delta features with nondelta features is performed. The best SI accuracy of 94% was achieved after the fusion of MFCC, delta MFCC and delta-delta MFCC with 18 feature vectors. While [52] showed how the accuracy of SI is increased by fusing models, a new type of generalized fuzzy model (GFM) was implemented and combined with GMM and the hidden markov model (HMM). The HMM-GFM combination achieves an accuracy of 93%. In [53], a three-step score fusion method was proposed and an SI system was tested with and without adding white gaussian noise (AWGN) and nonstationary noise (NSN) using MFCC, the power normalized cepstral coefficient (PNCC) and GMM-UBM acoustic modeling. The best SI accuracy of 95.83% was obtained when testing the clean speech data in NIST-2008 [53]. In [54], a comparison was made between the i-vector model and GMM-UBM on clean and noisy speech of 120 speakers from the TIMIT and NIST-2008 speech datasets with 7 types of score fusion techniques. The highest SI accuracy of 96.67% was achieved using the i-vector approach, while an SI accuracy of 95.83% was achieved using GMM-UBM on clean NIST-2008 data.

Research paper [55] used the i-vector and x-vector approaches and proposed attentive pooling for deep speaker embedding for a text-independent speaker verification (SV) system using the voxceleb1 and NIST-2012 voice datasets. Mainly four pooling techniques, including (i) simple average pooling, (ii) statistics pooling, (iii) attentive average pooling, and (iv) attentive statistics pooling, were used in [55]. From the experimental results, it was observed that the best EER of 3.85% was achieved using attentive statistics pooling (x-vector), and with the i-vector, the best EER of 5.39% was achieved when the voxceleb1 dataset was used for training and evaluation. In [56], a fully automated pipeline based on computer vision techniques was used on the voxceleb1 dataset from open-source media. Research paper [56] showed that a CNN-based architecture obtained the best result for an SI accuracy of 80.5% using the top1 classification accuracy, and the best EER of 7.8% was obtained for SV.

In [57], the use of genetic programming (GP) for feature selection in speaker verification systems is introduced. The three-step score fusion method is proposed in [58] for speaker identification (SI) systems. The system is tested with and without the addition of white gaussian noise (AWGN) and non-stationary noise (NSN) using features like MFCC and power-normalized cepstral coefficients (PNCC) with GMM-UBM acoustic modeling. In [59], a comparison is made between the i-vector and GMM-UBM models using clean and noisy speech from TIMIT and NIST-2008 speech data. In [60], the mathematical derivation shows that independent component analysis (ICA) can improve feature representation for non-Gaussian signals. A comparative study presented in [61] analyzes MFCC, IMFCC, LFCC, and PNCC speech features using gaussian mixture model (GMM) modeling for clean and noisy speech conditions. In [62], a new feature for speaker verification is proposed, which combines the advantages of low-variance multi-taper short-term spectral estimators and the acoustic robustness of gammatone filterbanks.

1.3 Contribution

The contributions of the proposed research can be categorized into two key areas:

1.3.1 Novel feature aggregation for enhanced speaker recognition

In the first publication, the research significantly advances speaker recognition through the introduction of a groundbreaking feature aggregation methodology. This innovative approach optimally utilizes 18 features, combining spectral and temporal speech features, including their delta and delta-delta values. Additionally, the research presents a versatile common feature fusion model tailored for different speech dataset sizes, demonstrating its adaptability and robustness through extensive experimentation.

The study further includes a comprehensive analysis of 315 unique feature fusion models on the NIST-2008 database, identifying the top 35 performing models. These models undergo evaluation on four additional datasets, showcasing their efficiency across diverse computational environments.

Furthermore, the research explores factors influencing speaker recognition performance, offering optimized feature fusion models for small, medium, and large-scale voice datasets. This thorough analysis emphasizes the importance of optimization in speaker recognition systems.

1.3.2 Feature optimization methods for efficient speaker recognition

In the second publication, the research introduces a series of feature optimization methods, including PCA-MPA, PCA-GA, ICA-MPA, ICA-GA, features-GA, and features-MPA. These methods are designed to enhance speaker recognition systems by identifying optimal feature combinations that improve accuracy while reducing the dimensionality of the feature space, leading to faster computation.

The study also places a specific focus on computational timing for both training and testing phases. By addressing these aspects, the research not only enhances speaker recognition accuracy but also makes it suitable for diverse datasets, expanding its potential use cases.

1.4 Thesis organization

The thesis is organized into distinct sections for clarity and coherence. The introduction provides comprehensive coverage of background information, related work, thesis contributions, and an overview of the overall organization. Chapter 2 delves into the methodology, focusing on feature-level fusion and the techniques applied for feature extraction. Chapter 3 elaborates on approach 2, explaining the dimension reduction technique and the associated optimization steps. Chapter 4 is dedicated to discussing approach 3, detailing the feature optimization method utilized. Chapter 5 explores the classification algorithm chosen for the study. The evaluation section in chapter 6 concentrates on database preparation, performance assessment for speaker identification and verification, the implementation of the ROC curve for analysis, results are scrutinized and compared across various approaches, encompassing feature-level fusion, dimension reduction, and feature optimization. This segment also includes discussions on system configuration, a comparative analysis with existing approaches, and an examination of computation time for training and testing, considering factors influencing speaker recognition performance. The conclusion in the final chapter 7 succinctly summarizes key findings and contributions while proposing avenues for future research. Additionally, a dedicated publications section lists the research output stemming from the presented thesis.

Chapter 2

Methodology for Feature-Level Fusion

2.1 Motivation

Many studies have been performed on feature-level fusion [40,49,51], but they mainly involve the fusion of MFCC features with other features and MFCC delta and delta–delta values. The proposed method, which uses fusion MFCC, LPC, PLP, RMS, centroid, and entropy information and combinations of their delta and delta–delta values to further improve the SR performance and to find a unique feature fusion model suitable for speech databases of various sizes, is implemented. The main advantage of using feature-level fusion is the recognition of correlated feature values produced by different biometric algorithms, thereby determining a compact set of relevant features that can enhance SR accuracy and remove redundant features to improve the SR results.

We propose three techniques for speaker recognition (SR) (figure 4), 1) feature fusion methodology, 2) dimension reduction, and 3) feature optimization using genetic algorithms (GA) and marine predator algorithm (MPA). The primary objective of our study is to compare these approaches and determine the best model for SR system. Through our comparative analysis, we aim to identify the strengths and limitations of each technique and evaluate their performance in the context of speaker recognition. By integrating these techniques, we can leverage their respective advantages and enhance the accuracy and efficiency of the speaker recognition system. Figure 4 shows total number of approaches used in this work.

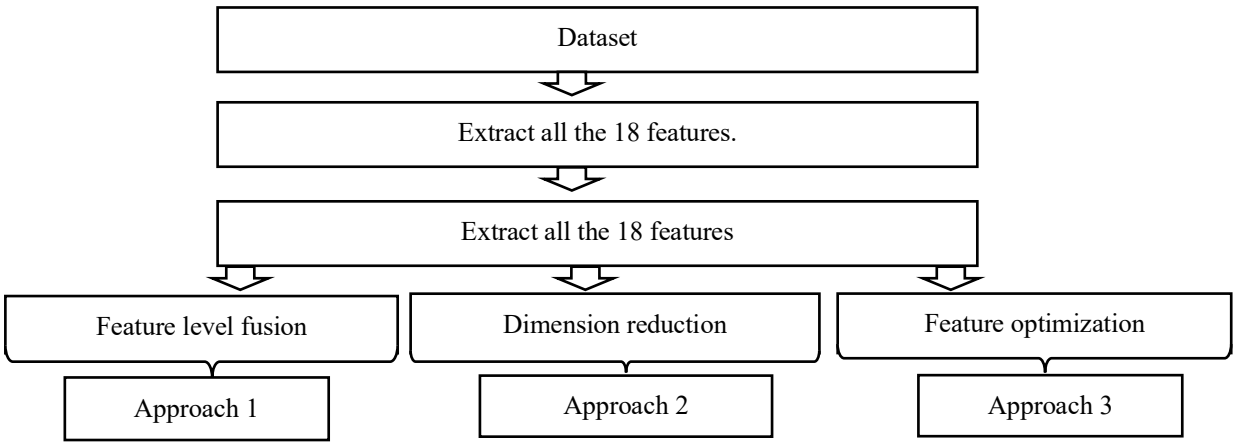


Fig.4 Proposed approach for speaker recognition model.

2.2 Feature fusion approach (Approach 1)

Automatic speaker recognition (ASR) is a state-of-the-art technique. Feature extraction and feature mapping are two important processes in SR. Feature extraction extracts several feature vectors called descriptors, and feature matching is used to avoid redundancy present in speech signal features and is used to compare the feature vectors extracted from a signal belonging to an unknown speaker with those extracted from a signal belonging to a known speaker set [4] .

Feature fusion is a method of combining features extracted from different sources or databases to create a single, more informative feature set. The Spectral features extracts frequency, power and other characteristics of a signal like (MFCC, LPC, PLP, centroid, entropy). Prosodic features are those features of speech that deal with the auditory properties of sound, such as stress, loudness variation, intonation (RMS). Systems based on spectral features, such as MFCC, perform better than prosody-based systems (pitch, rms feature) but their combined performance can provide the robustness needed for recognition systems .

2.3 Feature extraction

For the proposed work, following features and their delta and delta-delta values are used. Mirtoolbox [63] is used in matlab to compute the feature vectors of MFCC, centroid, RMS, and entropy.

2.3.1 Mel frequency cepstral coefficient (MFCC)

Since the mid-1980s, MFCCs have been the most popular method for feature extraction in automatic speech recognition (ASR) [35,37]. The extraction of MFCC feature vectors involves several steps, as outlined below.

1. Framing: Speech signals, being continuous in nature, are divided into frames of duration typically ranging from 20 to 40 milliseconds.
2. Windowing: Since speech signals are nonstationary and their parameters change approximately every 10 milliseconds, a windowing technique such as the hamming window is applied to the frames.
3. FFT (Fast Fourier Transform): The framed speech signals are transformed from the time-domain to the frequency-domain using the FFT algorithm.
4. Mel-filter bank Transformation: In this step, the transformed speech signal is passed through a bank of filters that are spaced according to the mel scale. The mel scale is derived based on the logarithmic perception of frequencies by the human auditory system. The mel scale (Mel(f)) can be calculated using equation (1):

$$\text{Mel}(f) = 2595 \log_{10}(1 + f/1000) \quad (1)$$

where f represents the actual frequency of the speech.

5. Logarithmic Conversion: The Mel-scaled frequencies obtained from the previous step are converted to a logarithmic scale. This conversion is linear up to 1 kHz and logarithmic for higher frequencies. The relationship between the frequency of speech and the mel scale can be established as (eq. 2):

$$\text{Frequency (mel scaled)} = [2695 \log (1 + f(\text{Hz}))/700] \quad (2)$$

6. Discrete Cosine Transform (DCT): This transformation de-correlates the speech features and arranges them in descending order of information. Typically, the first 13 DCT coefficients are selected as MFCC features [35] and [32].

2.3.2 Linear predictive coding (LPC)

LPC (Linear Predictive Coding) is widely used due to its speed, simplicity, and ability to capture time-varying formant information. It employs data encoding techniques to ensure secure transmission. Previous research [28, 35] has shown that combining cepstral features, including LPC, enhances speaker recognition (SR) results. Therefore, LPC is chosen as a feature extraction method in this work to improve recognition accuracy computes the current sample by combining past samples linearly. Inverse filtering is applied to remove formants from the speech signals, resulting in a residual signal known as the residue [65]. The VQ-LBG algorithm is used to calculate LPC features. For bitrate reduction, VQ is applied to LPC features in the linear spectral frequency (LSF) domain.

It is significant to recognize the autoregressive (AR) model of speech in order to understand LPCs. An audio signal can be modelled as a pth-order AR process, where each sample is given by Eqn. (3):

$$x(n) = -\sum_{k=1}^{p} a_k x(n-k) + u(n) \quad (3)$$

At the nth instant, each sample depends on 'p' previous samples, combined with gaussian noise u(n). The LPC coefficients are denoted by 'a.' To estimate these coefficients, the Yule-Walker equations are employed. Equation (4) represents the autocorrelation at lag 'l,' which is denoted by R(l) and is part of the autocorrelation function.

$$R(l) = a_0 + \sum_{n=1}^{N} (x(n)x(n-1)) \quad (4)$$

The Yule-Walker equations take their final form in equations (5) and (6).

$$\sum_{k=1}^{p} a_k R(l-k) = R(l) \quad (5)$$

$$\begin{bmatrix} R(0) & \cdots & R(p-1) \\ \vdots & \ddots & \vdots \\ R(p-1) & \cdots & R(0) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix} = - \begin{bmatrix} R(1) \\ \vdots \\ R(K) \end{bmatrix} \quad (6)$$

To obtain the LPC coefficients, the final solution is provided in Equation 7.

$$a = -R^{-1}r \quad (7)$$

In order to simplify the system, we have utilized only the first 13 LPC coefficients. Further information regarding the proposed LPC features, the VQ-LBG algorithm, and its calculation steps can be found in references [64] and [65].

2.3.3 Perceptual linear prediction (PLP)

In this study, PLP (perceptual linear prediction) features are chosen for their capability to effectively reduce noise, suppress reverberation, and eliminate echoes, leading to improved performance. Past research [28] have demonstrated that combining PLP features with cepstral features yields enhanced results in speaker recognition. The extraction of PLP features involves several steps, including equal loudness pre-emphasis, cube-root compression, and the removal of irrelevant speech information. The following steps outline the process to calculate PLP features.

- Following the application of the hamming window and FFT function to audio samples, converting them into the frequency domain, they are transformed into a power spectrum. Subsequently, this spectrum is mapped into a bark scale using the following approximation (eq 8):

$$\Omega(\omega) = 6 \cdot ln\left(\frac{\omega}{1200 \cdot \pi} + \sqrt{\left(\frac{\omega}{1200 \cdot \pi}\right)^2 + 1}\right) \quad (8)$$

where ω is the angular frequency in rad/s and $\boldsymbol{\Omega}$ represents the bark frequency.

- The bark-scaled spectra are integrated with the power spectra of the critical band filters. The ear's frequency resolution is considered constant on the bark scale. The resulting samples of the critical band power spectrum, approximated by the critical band curve $\Psi(\boldsymbol{\Omega})$, can be expressed as follows: eq (9).

$$\theta(\Omega_t) = \sum\nolimits_{\Omega} \left(p_{(\Omega - \Omega_t) \cdot \Psi(\Omega)}\right) \quad (9)$$

- Equal loudness pre-emphasis is employed to address the varying perception of loudness at different frequencies, and this compensation is achieved using the equation (10).

$$E\big(\Omega(\omega)\big) = E(\omega) \cdot \theta\big(\Omega(\omega)\big) \quad (10)$$

- In this context, $E(\boldsymbol{\Omega})$ serves as an approximation for the non-equal sensitivity of human hearing. Following that, the perceived loudness, denoted as $\Gamma(\boldsymbol{\Omega})$, is computed by taking the cube root of the intensity, adhering to the power law of hearing (Equation 11).

$$\Gamma(\Omega) = \sqrt[3]{E(\Omega)} \quad (11)$$

- In the last stage of PLP, $\Gamma(\Omega)$ is determined from the spectrum using the auto-correlation method of all-pole spectral modeling. Subsequently, the inverse DFT (IDFT) is applied to $\Gamma(\Omega)$ to obtain the autocorrelation function. These autoregressive coefficients can then be transformed into other sets of parameters of interest, such as cepstral coefficients.
- The detailed steps for extracting PLP features can be found in references [66,67]. Thirteen PLP features are computed by averaging all the PLP features for each voice, effectively reducing the system complexity using MATLAB software [58-59]. To ensure that the PLP feature dimension aligns with that of other features, the mean value of each frame is computed, resulting in 13x1 feature vectors per audio file. Figure 5 provides a visual representation of the feature extraction process for MFCC, LPC, and PLP features.

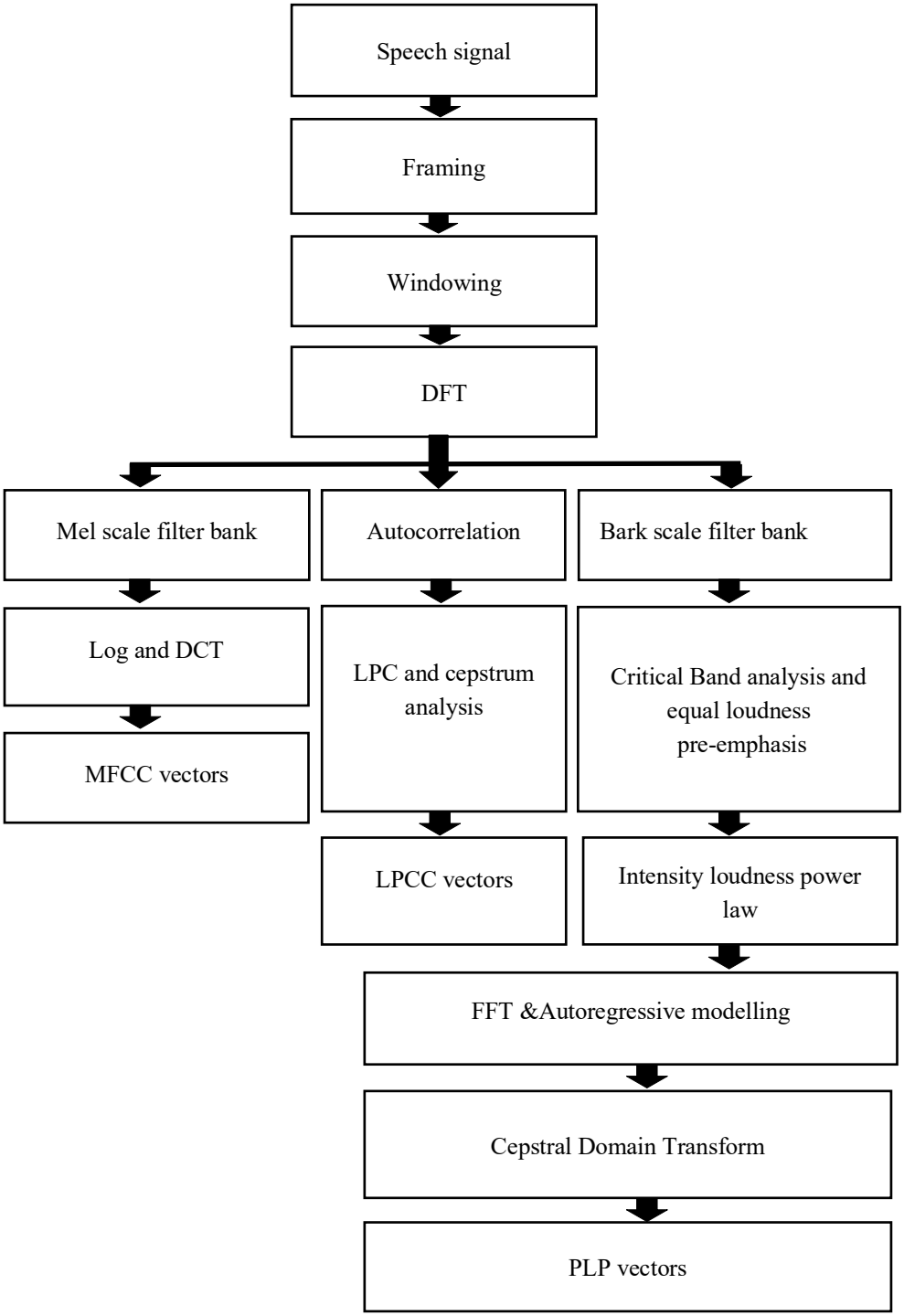


Fig.5 MFCC, LPC and PLP feature extraction steps.

2.3.4 Spectral centroid (SC)

The spectral centroid (SC) defines the centroid of gravity of the magnitude spectrum of the short-time fourier transform and provides a single value representing the frequency domain characteristic of a speech signal. A higher SC value corresponds to a greater energy in the signal [68]. It is computed as follows using equation (12):

$$C(i) = \frac{\sum_{k=0}^{N-1} k|x_i(k)|}{\sum_{k=0}^{N-1}|x_i(k)|} \quad (12)$$

2.3.5 Spectral entropy (SE)

Entropy spectral estimation is a method of estimating spectral density, computed as follows:

- Given a signal x(t), compute s(f), the power spectral density, by taking the fourier transform of the autocorrelation function of the signal x(t).
- Extract the power in the spectral band, depending on the frequency of interest. After calculating the spectral band power, normalize the power within the given band of interest.
- Calculate the spectral entropy using equation 13 [69].

$$SE = \sum s(f) * ln\frac{1}{s(f)} \quad (13)$$

2.3.6 Root mean square (RMS)

Root mean square (RMS) is a measure of the loudness of an audio signal. It is computed by taking the square root of the sum of the mean squares of the amplitudes of the sound samples. The RMS formula, given in Equation 14 [70], can be expressed as follows: where x1, x2,…xn represent n observations, and xrms denotes the RMS value for the n observations.

$$x_{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x\, i^2} \quad (14)$$

2.3.7 Delta features

Delta features are essential for capturing the rate of change in speech features, particularly the power dynamics of speech signals concerning noise. By incorporating delta (Δ) and delta-delta (ΔΔ) features, valuable dynamic information can be extracted from speech signals [16],[51]. The computation of the delta feature Δk involves subtracting the current feature fk from the previous feature fk-1, as represented in Equation (15):

$$\Delta k = fk - fk\text{-}1 \quad (15)$$

Likewise, the delta-delta feature ΔΔk is obtained by subtracting the current delta feature Δk from the previous delta feature Δk-1, as depicted in Equation (16):

$$\Delta\Delta k = \Delta - \Delta k\text{-}1 \quad (16)$$

Table 1 provides the dimensions of each feature utilized in the study.

Table 1 Feature dimension.

Feature	Number of feature vector for 1 audio file (Row x Column)
MFCC, ΔMFCC, ΔΔMFCC	13x1,13x1,13x1
LPC, ΔLPC, ΔΔLPC	13x1,13x1,13x1
PLP, ΔPLP, ΔΔPLP	13x1,13x1,13x1
Centroid, ΔCentroid, ΔΔCentroid	1x1,1x1,1x1
RMS, ΔRMS, ΔΔRMS	1x1,1x1,1x1
Entropy, ΔEntropy, ΔΔEntropy	1x1,1x1,1x1

2.4 Mirtoolbox

MIRtoolbox is a comprehensive matlab toolbox that specializes in extracting musical features from audio files. Its functionalities include a range of routines for statistical analysis, segmentation, and clustering. The toolbox's modular design is based on a philosophy of leveraging expertise: techniques initially developed for specific domains of music analysis are

transformed into general operators that can be applied to various analytical contexts. Notably, each feature extraction method within MIRtoolbox can be employed with an audio file as an argument or with any interim results from intermediary stages of the operational chain. Moreover, the same syntax is adaptable for analyzing single audio files, batches of files, sequences of audio segments, multi-channel signals, and more. To support this versatility, the toolbox organizes its data and methods within an object-oriented architecture [63]. Figure 6 shows type of speech features calculated using MIRtoolbox.

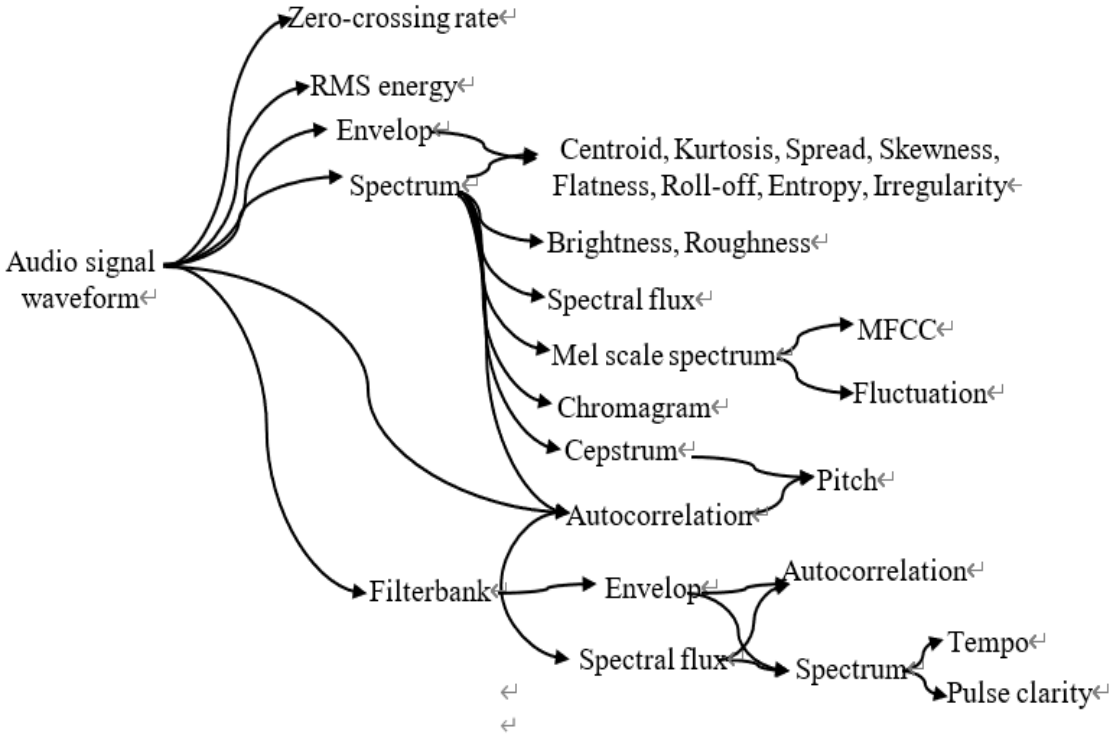


Fig. 6 Overview of the musical features that can be extracted with MIRTOOLBOX.

2.5 Audacity software

We utilized the audacity software, a widely-used, free, and open-source digital audio editor and recording application. This software is accessible across various operating systems such as windows, macOS, linux, and other unix-like systems. As of December 6, 2022, audacity holds the distinction of being the most frequently downloaded software on FossHub, boasting an impressive download count of over 114.2 million since March 2015. We leveraged the capabilities of audacity to segment the voice into distinct parts for our project. After recording your podcast, utilize audacity for editing. You can import additional audio files for a seamless edit. Remember that audacity is an audio editing software, not a DAW, and changes are irreversible upon saving. The Selection tool in the tools toolbar functions like a word processor cursor, allowing you to select segments for processing. For instance, you can cut, paste, or delete highlighted segments, and audacity automatically adjusts the surrounding clips accordingly. Figure 7 shows segmentation of speech using audacity software.

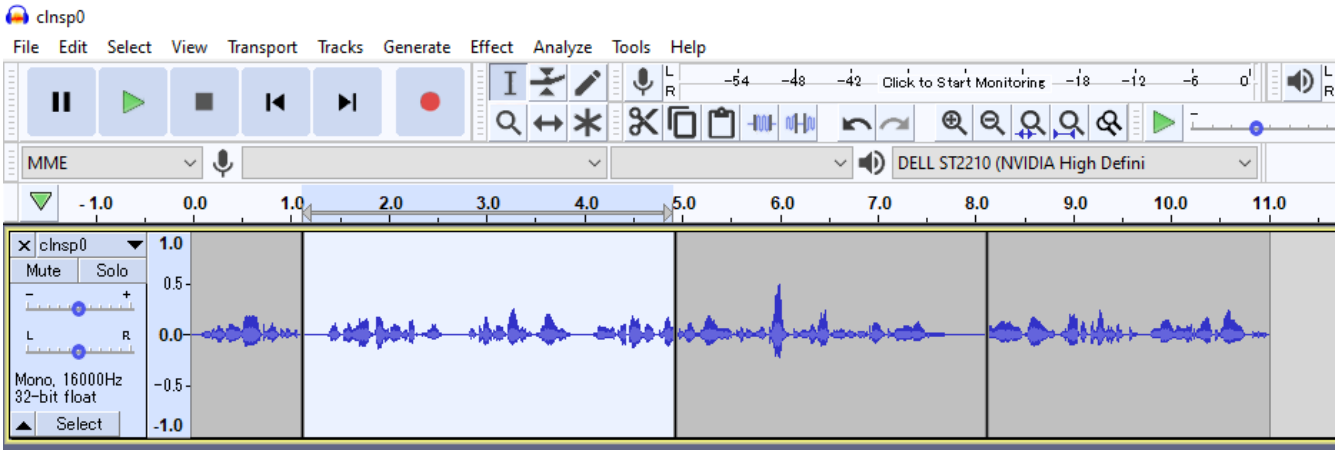


Fig.7 Speech segmentation using audacity software.

2.6 Programming

2.6.1 Matlab code for MFCC feature extraction

```
for i1= 1:10
   disp(i1)
   filename1 = sprintf('s1%d.wav',i1);
   [signal1,fs1] = audioread(filename1);
   mfcc1(:,i1) = mirmfcc(filename1);
end
```

2.6.2 Matlab code for RMS feature extraction

```
 for i1= 1:10
   disp(i1)
   filename1 = sprintf('s1%d.wav',i1);
   [signal1,fs1] = audioread(filename1);
   mfcc1(:,i1) = mirrms(filename1);
end
```

2.6.3 Matlab code for entropy feature extraction

```
for i1= 1:10
   disp(i1)
   filename1 = sprintf('s1%d.wav',i1);
   [signal1,fs1] = audioread(filename1);
   mfcc1(:,i1) = mirmfcc(filename1);
end
```

2.6.4 Matlab code for centroid feature extraction

```
for i1= 1:10
```

```
    disp(i1)
    filename1 = sprintf('s1%d.wav',i1);
    [signal1,fs1] = audioread(filename1);
    centroid1(:,i1) = mircentroid(filename1);
end
```

Note: (This script is designed to process 10 voice samples from speaker 1, calculate their MFCCs, rms, entropy, centroid and store the results in the matrix `mfcc1, rms1, entropy1, centroid1 ` for further analysis.)

2.6.5 Feature extraction with mirtoolbox: A comprehensive code explanation

In this section, we provide a detailed explanation of the code implementation for feature extraction using MIRtoolbox.

MFCC, rms, entropy, centroid is calculated using mirtoolbox. Below is the explanation of the function used.

1. **for i1 = 1:10:** A loop that iterates 10 times, with the loop variable `i1`. This loop is designed to iterate over 10 voice samples of speaker 1.
2. **disp(i1):** Displays the current value of the loop index `i1`. This is likely used to monitor the progress of the loop during execution.
3. **filename1 = sprintf('s1%d.wav', i1):** Generates a file name for each voice sample by combining the string 's1' with the value of i1 and .wav.
4. **[signal1, fs1] = audioread(filename1):** Reads the audio file specified by `filename11` and stores the audio data in signal1. The sampling frequency is stored in fs1.
5. **mfcc1(:, i1) = mirmfcc(filename1), rms1(:, i1) = mirrms(filename1), entropy1(:, i1) = mirentropy(filename1), centroid1(:, i1) = mircentroid(filename1) :**Computes the mel-frequency cepstral coefficients (MFCC),root mean square (RMS),centroid and entropy features of the audio signal loaded from the file specified by `filename1` and stores the result in the matrix ‘mfcc1,rms1,centroid1,entropy1` at column i1 respectively.

2.6.6 Matlab code for linear predictive coding (LPC) feature extraction

```
for i1 = 1:10
    disp(i1);
    filename1 = sprintf('s1%d.wav',i1);
    [s1,fs1]= audioread(filename1);
    t1 = [240 400 0];   % sampling frequency was 16kHz, frame size was 25ms. Hence the number of samples were 400.
Overlapping 40%
    p1=13;
    c1 = lpcauto(s1,p1,t1);
    v1 = c1(1,:)';
    k1 = 1; % number of centroid required
    r1 = vqlbg(v1,k1);
    lpc1(:,i1) = r1(2:14);
end
```

2.6.7 LPC code Explanation

The provided MATLAB code processes 10 audio files, performs linear predictive coding (LPC) analysis on each of them, and stores the results for further analysis. Here is a summarized breakdown of each line:

1. **for i1 = 1:10:** Initiates a loop that will iterate 10 times, with the loop index variable named `i1`.
2. **disp (i1):** Displays the current value of the loop index `i1`.

3. **filename1 = sprint ('s1%d.wav', i1):** Generates a string containing the file name by combining the string 's1' with the current value of `1` and '.wav' to create a filename.
4. **[s1,fs1]= audioread(filename1):** Reads the audio file specified by `filename1` using the `audioread` function. It stores the audio data in the variable `s1` and the sampling frequency in `fs1`.
5. **t1 = [240 400 0]:** Defines a vector `t1` related to parameters of the audio processing, such as frame size, sampling frequency, and the percentage of overlap.
6. **p1=13:** Initializes the variable `p1` to the value 13, likely a parameter for use in the subsequent function.
7. **c1 = lpcauto(s1,p1,t1):** Computes the linear predictive coding (LPC) coefficients of the audio signal `s1` using the parameters `p1` and `t1`.
8. **v1 = c1(1,:):** Extracts specific data from `c1` and stores it in `v1`.
9. **k1 = 1:** Sets the variable `k1` to the value 1, indicating the number of centroid required.
10. **r1 = vqlbg(v1,k1):** Applies vector quantization using the Linde-Buzo-Gray (LBG) algorithm to the vector `v1` with the specified number of centroid `k1`.
11. **lpc1(:,i1) = r1(2:14):** Assigns a portion of the data from `r1` to the matrix `lpc1` for further analysis. The data from indices 2 to 14 of `r1` are stored in column `i1` of `lpc1`.

2.6.8. Matlab code for PLP feature extraction

```
for i1= 1:10
   disp(i1)
   filename1 = sprintf('s11%d.wav',i1);
  [s1,fs1]= audioread(filename1);
[cepp1, specp1] = rastaplp(s1, fs1, 0, 12);
plp3(:,i1) = mean (cepp1,2);
end
```

NOTE: Download code from https://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/

2.6.9 Code explanation for PLP

1. **for i1 = 1:10**: Initiates a loop that will run 10 times, with the loop index variable named `i1`.
2. **disp(i1):** Displays the current value of the loop index `i1`.
3. **filename1 = sprintf('s11%d.wav',i1)**: Generates a string containing the file name by combining the string 's11' with the current value of `i1` and '.wav'.
4. **[s1,fs1]= audioread(filename1)**: Reads the audio file specified by `filename1` using the `audioread` function. It stores the audio data in the variable `s1` and the sampling frequency in `fs1`.
5. **[cepp1, specp1] = rastaplp(s1, fs1, 0, 12):** Computes the RASTA-PLP (Rasta Perceptual Linear Prediction) cepstral coefficients and the log power spectrum of the audio signal `s1` using the sampling frequency `fs1` and the specified parameters 0 and 12.
6. **plp3(:,i1) = mean(cepp1, 2)**: Computes the mean of the RASTA-PLP cepstral coefficients along the second dimension (columns) and stores the result in the matrix `plp3` at column `i1`.

In summary, this code processes 10 audio files, calculates the RASTA-PLP cepstral coefficients, takes their mean, and stores the results in the matrix `plp3` for further analysis or processing.

2.6.10 Delta and delta-delta feature calculation matlab code

```
del = deltas(mfcc);
% Double deltas are deltas applied twice with a shorter window
ddel = deltas(deltas(mfcc, 5), 5);
```

Note replace mfcc with rms, centroid, entropy, lpc, plp for calculation of delta and delta-delta values .

2.6.11 Code explanation for delta and delta-delta feature calculation

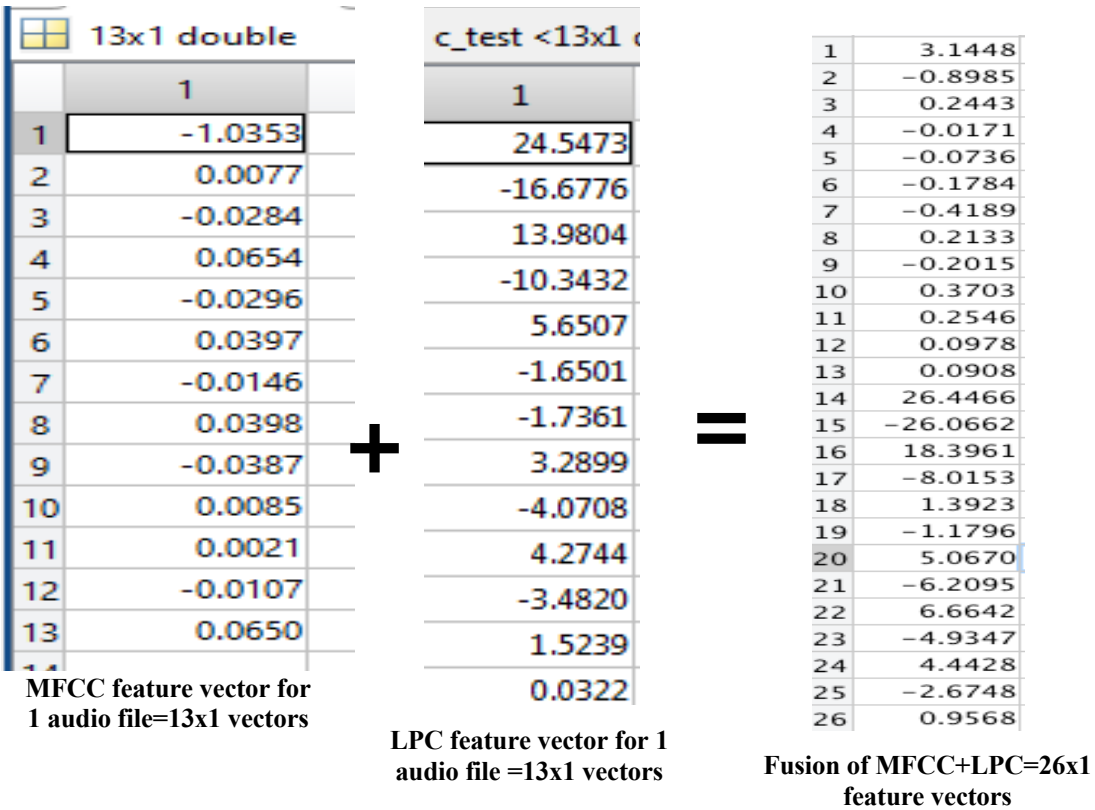
1. **del = deltas(mfcc):** Computes delta features, capturing the rate of change of MFCCs over time, commonly used in speech and audio processing.
2. **ddel = deltas(deltas(mfcc, 5), 5):** Computes double delta features by applying deltas twice with a shorter window, providing information about acceleration or curvature of MFCCs over time.

In summary, the code calculates delta and double delta features from MFCCs, enhancing temporal information for speech and audio processing.

2.7 Feature fusion methodology

Feature fusion is a method of combining features extracted from different sources or databases to create a single, more informative feature set. The spectral features extracts frequency, power and other characteristics of a signal like (MFCC, LPC, PLP, centroid, entropy). Prosodic features are those features of speech that deal with the auditory properties of sound, such as stress and RMS. Systems based on spectral features, such as MFCC, perform better than prosody-based systems (pitch, rms feature) their combined performance can provide the robustness needed for recognition systems .

In our research, we've maintained uniform feature length and dimension to simplify feature fusion. We utilized the mirtoolbox in matlab, a specialized toolbox for audio feature extraction. Fig. 8 shows the example for fusion of two features using .mat files in matlab.



MFCC feature vector for 1 audio file=13x1 vectors

LPC feature vector for 1 audio file =13x1 vectors

Fusion of MFCC+LPC=26x1 feature vectors

Fig.8 Screenshot from matlab for the fusion of 2 feature using .mat files.

2.7.1 Optimization steps for features fusion

Following steps shows how different features are fused and how best models are selected [1].
Table 2-table19 shows the SI accuracy result for each fusion steps. Model highlighted with red are the 2 best model selected at each fusion steps. MFCC, LPC, PLP, centroid, RMS, entropy, and their delta and delta-delta features vectors are extracted for all 5 voice datasets using MATLAB software. Table 1 shows total number of feature vectors extracted for one audio file.

The input matrix files are created using single features and combinations of features in MATLAB. The labeling of each speaker is done under the value of their features to do testing using classification learner application.
Following steps shows the methodology of the proposed feature fusion approach.

1. **Steps 1**: The first step of feature fusion involves training and testing all 18 features individually and then selecting the best 2 features with the highest SI accuracy and lowest average EER among all 18 features. To select the best model, the average accuracy and average EER values of the three classifiers are considered. PLP and MFCC are the first- and second-best models, respectively, because they have the highest average accuracy values of 80% and 61.1% and the lowest EER values of 5.2% and 15.4%, respectively, compared to other features. Equation 17 shows the calculation of the average accuracy and EER values using all three classifier results. Table 2 shows the result of speaker identification accuracy for all 18 features.

$$\text{Average result} = \frac{\text{LD+KNN+ensemble (accuracy or EER)}}{3} \quad (17)$$

Table 2 Speaker identification accuracy for all 18 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
MFCC	72.5	42.7	68.1
Δ(MFCC)-1	14.6	36.7	40
ΔΔ(MFCC)	13.5	19	24.2
LPC	57.7	63.1	56.9
ΔLPC	13.1	45.2	37.5
ΔΔLPC	29.2	56.5	41.5
PLP	85.6	73.5	82.9
ΔPLP-2	15.6	49.2	15.2
ΔΔPLP	36	36.9	22.3
centroid	5.5	3.8	4.8
Δcentroid	2.1	2.1	2.1
ΔΔcentroid	2.5	2.5	2.7
entropy	8.3	8.3	5
Δentropy	0.8	1.9	1.9
ΔΔentropy	2.9	2.9	3.1
RMS	7.7	7.1	7.7
ΔRMS	1.2	1.5	3.3
ΔΔRMS	2.9	1.9	2.9
MFCC	72.5	42.7	68.1

2. **Steps 2:** In the second step, 2 features are fused by combining the best features, MFCC and PLP, separately with the remaining 17 features, and again, the best two models are selected from this step. The two best models from this step are the MFCC and Δentropy fusion model and the PLP and LPC fusion model. Table 3 shows the SI accuracy for the fusion of 2 features.

Table 3 Speaker identification accuracy using fusion of 2 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+MFCC	86.9	78.5	86.5
PLP+ΔMFCC	88.5	78.1	90
PLP+ΔΔMFCC	84.4	73.1	80.4
PLP+LPC	89.8	87.1	88.8
PLP+ΔLPC	78.8	84	88.3
PLP+ΔΔLPC	84	85.6	80.2
PLP+ΔPLP	86.7	85.6	83.3
PLP+ΔΔPLP	88.8	72.3	85.2
PLP+RMS	85.4	78.8	85.6
PLP+ΔRMS	85.2	77.1	81.7

PLP+ΔΔRMS	83.8	71.7	81
PLP+centroid	60	60.9	81.6
PLP+Δcentroid	50	45	81
PLP+ΔΔcentroid	45	45	80
PLP+entropy	84.4	76	84
PLP+Δentropy	87.5	77.1	85
PLP+ΔΔentropy	83.8	76.2	81.5
MFCC+ΔMFCC	77.1	57.9	74.6
MFCC+ΔΔMFCC	73.8	37.5	69.4
MFCC+LPC	85.4	73.1	83.5
MFCC+ΔLPC	61.7	72.9	65.6
MFCC+ΔΔLPC	84.4	82.7	81.5
MFCC+PLP	85.2	75	84.2
MFCC+ΔPLP	75.8	67.1	89.2
MFCC+ΔΔPLP	70.6	50.4	70.4
MFCC+RMS	70.5	46.5	70.2
MFCC+ΔRMS	73.8	42.5	71.2
MFCC+ΔΔRMS	71.5	42.1	71.5
MFCC+centroid	45	37.3	67.1
MFCC+Δcentroid	30	25.4	69
MFCC+ΔΔcentroid	45	23.3	66.2
MFCC+entropy	73.3	45.4	73.1
MFCC+Δentropy	85.2	84.2	89
MFCC+ΔΔentropy	74.6	37.9	72.1

3.Step 3: In the third step, 3 features are fused by combining the remaining 16 features separately with the two best models selected from step 2. Two best model from this step are PLP+LPC+ΔΔLPC and PLP+LPC+ΔPLP (Table 4).

Table 4 Speaker identification accuracy using fusion of 3 features.

Features	**Speaker identification accuracy**		
	LD	**KNN**	**Ensemble**
PLP+LPC+MFCC	90.6	84.4	90.6
PLP+LPC+ΔMFCC	91.5	89	90.6
PLP+LPC+ΔΔMFCC	88.8	84.8	88.3
PLP+LPC+ΔLPC	86.9	95.2	86.7
PLP+LPC+ΔΔLPC	93.1	93	90.8
PLP+LPC+ΔPLP	92.3	95.2	90.6
PLP+LPC+ΔΔPLP	89.6	84.2	87.3
PLP+LPC+RMS	90.4	87.5	88.8
PLP+LPC+ΔRMS	90.4	85.4	89.6
PLP+LPC+ΔΔRMS	89.8	88.8	87.5
PLP+LPC+centroid	66	67.3	89.4
PLP+LPC+Δcentroid	55	51	89.2
PLP+LPC+ΔΔcentroid	50	48.8	89.6
PLP+LPC+entropy	90.6	87.3	88.5
PLP+LPC+Δentropy	92.3	86.5	90.6
PLP+LPC+ΔΔentropy	92.9	85.6	91.5
MFCC+Δentropy+ΔMFCC	73.1	55	72.9
MFCC+Δentropy+ΔΔMFCC	76.2	40.4	74.2
MFCC+Δentropy+LPC	82.7	76.5	82.5
MFCC+Δentropy+ΔLPC	67.5	65	65
MFCC+Δentropy+ΔΔLPC	73.2	71	70.4
MFCC+Δentropy+PLP	85.8	78.1	86.5

MFCC+Δentropy+ΔPLP	75.2	68.5	74.6
MFCC+Δentropy+ΔΔPLP	72.7	51.2	71.5
MFCC+Δentropy+RMS	73.7	50	70.7
MFCC+Δentropy+ΔRMS	73.1	40.6	70.8
MFCC+Δentropy+ΔΔRMS	73.5	38.3	70.4
MFCC+Δentropy+centroid	44	35	66.9
MFCC+Δentropy+Δcentroid	40	25	68.3
MFCC+Δentropy+ΔΔcentroid	35	25.8	68.4
MFCC+ΔentropY+entropy	69.8	42.9	68.5
MFCC+Δentropy+ΔΔentropy	76	35.6	72.5

4.Step 4: Fusion of 4 features are done by combining PLP+LPC+ΔΔLPC and PLP+LPC+ΔPLP feature fusion model with the remaining feature and best model is selected are PLP+LPC+ΔPLP+MFCC and PLP+LPC+ΔPLP+ΔΔLPC (Table 5).

Table 5 Speaker identification accuracy using fusion of 4 features.

Features	**Speaker identification accuracy**		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔΔLPC+MFCC	92.9	87.7	91.9
PLP+LPC+ΔΔLPC+ΔMFCC	93.1	89.8	90.8
PLP+LPC+ΔΔLPC+ΔΔMFCC	93.1	81.5	90
PLP+LPC+ΔΔLPC+ΔLPC	91.2	92.5	88.8
PLP+LPC+ΔΔLPC+ΔPLP	95	94	90.8
PLP+LPC+ΔΔLPC+ΔΔPLP	95.8	82.1	92.5
PLP+LPC+ΔΔLPC+RMS	89.6	86.2	88.3
PLP+LPC+ΔΔLPC+ΔRMS	92.3	89.8	89.8
PLP+LPC+ΔΔLPC+ΔΔRMS	92.1	85.2	89.2
PLP+LPC+ΔΔLPC+centroid	60	66	86.9
PLP+LPC+ΔΔLPC+Δcentroid	50	50	90.2
PLP+LPC+ΔΔLPC+ΔΔcentroid	50	49.4	89
PLP+LPC+ΔΔLPC+entropy	93.1	85.8	89
PLP+LPC+ΔΔLPC+Δentropy	93.3	86	89
PLP+LPC+ΔΔLPC+ΔΔentropy	93.5	87.5	90.4
PLP+LPC+ΔPLP+MFCC	93.3	95.2	93.1
PLP+LPC+ΔPLP+ΔMFCC	90.2	93.3	89
PLP+LPC+ΔPLP+ΔΔMFCC	91.5	91	88.8
PLP+LPC+ΔPLP+ΔLPC	90	96.5	88.5
PLP+LPC+ΔPLP+ΔΔLPC	93.3	94.4	91.9
PLP+LPC+ΔPLP+ΔΔPLP	93.5	92.3	90.8
PLP+LPC+ΔPLP+RMS	91.7	95.4	90
PLP+LPC+ΔPLP+ΔRMS	91.9	94.6	90
PLP+LPC+ΔPLP+ΔΔRMS	89.8	94.8	89.4
PLP+LPC+ΔPLP+centroid	73.5	73.5	84.6
PLP+LPC+ΔPLP+Δcentroid	50	50.4	87.3
PLP+LPC+ΔPLP+ΔΔcentroid	50	51.2	92.7
PLP+LPC+ΔPLP+entropy	94.8	95.2	93.1
PLP+LPC+ΔPLP+Δentropy	90.6	94.6	89.2
PLP+LPC+ΔPLP+ΔΔentropy	91.5	94	89.4

5.Step 5. Fusion of 5 features are done using best model PLP+LPC+ΔPLP+MFCC and PLP+LPC+ΔPLP+ΔΔLPC from step 4 and combining it with remaining features (table 6). Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP and PLP+LPC+ΔPLP+ΔΔLPC+RMS (table 6).

Table 6 Speaker identification accuracy using fusion of 5 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+LPC+ΔPLP+MFCC+ΔMFCC	92.1	95.4	91.7
PLP+LPC+ΔPLP+MFCC+ΔΔMFCC	91	89	89
PLP+LPC+ΔPLP+MFCC+ΔLPC	94.4	93.5	93.3
PLP+LPC+ΔPLP+MFCC+ΔΔLPC	92.7	93.1	91.7
PLP+LPC+ΔPLP+MFCC+ΔΔPLP	93.8	80.2	92
PLP+LPC+ΔPLP+MFCC+RMS	93.1	94.6	92.5
PLP+LPC+ΔPLP+MFCC+ΔRMS	91.9	95.4	91.7
PLP+LPC+ΔPLP+MFCC+ΔΔRMS	91.5	91.7	94
PLP+LPC+ΔPLP+MFCC+centroid	70	74.4	91.2
PLP+LPC+ΔPLP+MFCC+Δcentroid	50	52.3	90.4
PLP+LPC+ΔPLP+MFCC+ΔΔcentroid	52	52.7	91.2
PLP+LPC+ΔPLP+MFCC+entropy	93.3	94.8	91.2
PLP+LPC+ΔPLP+MFCC+Δentropy	90	93.1	89
PLP+LPC+ΔPLP+MFCC+ΔΔentropy	92.3	94.8	91.7
PLP+LPC+ΔPLP+ΔΔLPC+ΔMFCC	91.1	87.8	88
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔMFCC	92.1	88.8	89
PLP+LPC+ΔPLP+ΔΔLPC+ΔLPC	89.6	93.5	85.8
PLP+LPC+ΔPLP+ΔΔLPC+MFCC	94	92.1	92.1
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP	95.6	95	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS	95.6	92.7	93.8
PLP+LPC+ΔPLP+ΔΔLPC+ΔRMS	93.3	92.7	92.3
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔRMS	90.4	92.7	87.7
PLP+LPC+ΔPLP+ΔΔLPC+centroid	70	70	87.5
PLP+LPC+ΔPLP+ΔΔLPC+Δcentroid	49	49.4	85.4
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔcentroid	50	51.5	91.2
PLP+LPC+ΔPLP+ΔΔLPC+entropy	93.3	92.3	89.6
PLP+LPC+ΔPLP+ΔΔLPC+Δentropy	94.4	93.3	91.5
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔentropy	93.3	94	90

6.Step 6: Fusion of 6 features are done using best model PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP and PLP+LPC+ΔPLP+ΔΔLPC+RMS (table 6) from step 5 and combining it with remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC (table 7).

Table 7 Fusion of 6 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+MFCC	95.6	91.7	94
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔMFCC	95.4	91.5	94
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔΔMFCC	93.1	88.3	87.1
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔLPC	93.5	92.5	89.8
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+RMS	96.5	91.2	92.1
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS	95.8	91.5	93.3
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔΔRMS	94.8	92.3	90.6
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+centroid	70	70.2	89.4
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+Δcentroid	50	50	92.7
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔΔcentroid	50	51	91
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+entropy	94.8	88.3	91.2
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+Δentropy	94.2	90.2	89.8
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔΔentropy	94.2	90.4	90
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC	96	94.8	92.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔMFCC	93.1	93.8	90.8

PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔΔMFCC	93.8	91	90.4
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔLPC	91	92.7	89
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔΔPLP	94	93.1	91.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔRMS	93.8	94.2	90.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔΔRMS	93.8	92.7	90.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+centroid	70	70.2	86.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+Δcentroid	52	52.3	88.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔΔcentroid	50	51.9	90
PLP+LPC+ΔPLP+ΔΔLPC+RMS+entropy	92.5	94	89.4
PLP+LPC+ΔPLP+ΔΔLPC+RMS+Δentropy	93.8	93.5	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+ΔΔentropy	93.5	93.5	90.4

7.Step 7: Fusion of 7 features are done using best model PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC (table 7) from step 6 and combining it with remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS (table 8).

Table 8 Fusion of 7 features.

Features	**Speaker identification accuracy**		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+MFCC	92.3	90	91
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+ΔMFCC	93.3	91.7	91.5
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+ΔΔMFCC	94.2	87.1	92.3
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+ΔLPC	91.2	94.4	89.2
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+RMS	94.2	88.3	90.6
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+ΔΔRMS	95.2	91.7	91.2
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+centroid	67	67.7	88.8
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+Δcentroid	50	49.4	87.9
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+ΔΔcentroid	50	51.2	92.9
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+entropy	92.7	91.5	91
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+Δentropy	94.2	92.9	90.4
PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS+ΔΔentropy	94.2	93.3	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔMFCC	95.2	92.3	94.4
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC	96.2	90.8	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔLPC	94.4	95.6	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔRMS	94	92.9	90.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS	96	93.8	94.8
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+centroid	70	70.6	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+Δcentroid	50	50.6	90.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔcentroid	50	50	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+entropy	94.4	95.2	91.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+Δentropy	96	92.5	92.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔentropy	94.4	93.8	90.8

8. Step 8: Fusion of 8 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS (table 8) from step 7 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP (table 9).

Table 9 Fusion of 8 features.

Features	**Speaker identification accuracy**		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔMFCC	93.3	91	90.2

PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔLPC	95.2	94	91.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔΔPLP	96.2	90.8	94.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS	96.5	91.9	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔΔRMS	95.8	90.4	95
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+centroid	69	51.9	91.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+Δcentroid	69	69.6	92.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔΔcentroid	48	48.8	91.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+entropy	95.6	91.2	93.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+Δentropy	95.6	92.1	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔΔentropy	95.6	91.5	94
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔMFCC	93.8	94.2	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔLPC	93.5	94.2	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP	96.5	91.9	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔRMS	94.8	93.5	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔMFCC	94.4	91.5	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+centroid	74	74	92.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+Δcentroid	50	52.1	94.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔcentroid	50	52.2	94.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+entropy	94	93.1	92.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+Δentropy	94.4	95.2	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔentropy	94	91.7	93.1

9.Step9: Fusion of 9 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP ΔΔRMS (table 8) from step 8 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔLPC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔLPC (table 10).

Table 10 Fusion of 9 features.

Features	**Speaker identification accuracy**		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔMFCC	95.6	90.2	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔLPC	96.7	95	93.8
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔPLP	95.2	88.8	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔRMS	96.5	91	92.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+centroid	69	69.2	91.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δcentroid	49	49.6	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔcentroid	50	50.4	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+entropy	96.9	90.8	94.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δentropy	95.2	92.1	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔentropy	95.8	91.7	92.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC	95.4	92.9	91.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC	96.2	89	94.4
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔLPC	96	94.4	93.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS	96.5	92.1	94.8
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid	72	72.3	91.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid	70	69.3	90.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid	72	71.3	91.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+entropy	94.8	91.9	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ Δentropy	95.2	90.6	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy	96.2	94	94.6

10.Step10: Fusion of 10 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔLPC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔLPC (table 10).from step 9 with the remaining features.

Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔMFCC+ΔLPC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+entropy+ΔLPC (table 11).

Table 11 Fusion of 10 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔMFCC+ΔLPC	96.2	93.3	95
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔPLP+ΔLPC	96.2	89.6	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔRMS+ΔLPC	95	94.8	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+centroid+ΔLPC	70	71.5	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δcentroid+ΔLPC	68	71.5	91.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔcentroid+ΔLPC	70	69.5	90
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+entropy+ΔLPC	95	92.7	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δentropy+ΔLPC	94.4	93.1	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔentropy+ΔLPC	93.8	93.3	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC+ΔLPC	95.8	93.5	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC	95.8	92.7	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC	94.2	94	89.8
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC	70	71.5	92
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC	68	70.5	91.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC	70	69.5	90
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+entropy+ΔLPC	96.2	95.2	94.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ Δentropy+ΔLPC	93.3	94.2	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropY+ΔLPC	95.4	94	92.7

11.Step 11. Fusion of 11 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔMFCC+ΔLPC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+entropy+ΔLPC (table 11). From step 10 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔRMS+ΔLPC+ΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ Δentropy+ΔLPC+entropy ΔLPC (table 12).

Table 12 Fusion of 11 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔPLP+ΔLPC+ΔMFCC	95.6	92.1	94.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔRMS+ΔLPC+ΔMFCC	96.5	93.1	94.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+centroid+ΔLPC+ΔMFCC	70	70.4	89.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δcentroid+ΔLPC+ΔMFCC	49	49.8	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔcentroid+ΔLPC+ ΔMFCC	50	50.2	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+entropy+ΔLPC+ΔMFCC	92.7	92.7	89.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δentropy+ΔLPC+ΔMFCC	94.2	92.1	92.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔentropy+ΔLPC+ ΔMFCC	95.2	94.4	93.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC+ΔLPC+entropy	93.8	93.1	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy	96	93.1	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy	94.4	94	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC+entropy	50	52	91.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC+entropy	51	50	90.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC+entropy	50	50	91.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δentropy+ΔLPC+entropy	96.9	95	96
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy	94	93.3	91.5

12.Step 12: Fusion of 12 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔRMS+ΔLPC+ΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ Δentropy+ΔLPC+entropy ΔLPC (table 12) from step 11 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC+ΔLPC+entropy+ Δentropy and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy+ Δentropy (table 13).

Table 13 Fusion of 12 features.

Features	Speaker identification accuracy		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔPLP+ΔLPC +ΔMFCC+ΔΔRMS	95.4	89.8	93.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+centroid+ΔLPC +ΔMFCC+ΔΔRMS	68	68.3	91.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δcentroid+ΔLPC +ΔMFCC+ΔΔRMS	48	48.8	90.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔcentroid+ΔLPC +ΔMFCC+ΔΔRMS	51	51.9	93.8
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+entropy+ΔLPC +ΔMFCC+ΔΔRMS	95.8	91.9	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+Δentropy+ΔLPC +ΔMFCC+ΔΔRMS	95	94.2	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔentropy+ΔLPC +ΔMFCC+ΔΔRMS	94	93.3	91.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC+ΔLPC +entropy+ Δentropy	95.6	93.3	93.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC +ENTROPY+ Δentropy	95.2	92.3	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC +entropy+ Δentropy	96.9	95.6	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC +entropy+ Δentropy	70	70.6	91.2
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy	50	50	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ Δentropy	50	49.5	91.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy	93.8	94.6	93.8

13.Step 13: Fusion of 13 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC+ΔLPC+entropy+ Δentropy and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy+ Δentropy (table 13) from step 12 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy+ Δentropy+ΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy+ Δentropy+ΔMFCC (table 14).

Table 14 Fusion of 13 features.

Features	Speaker identification accuracy		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC +entropy+ Δentropy+ΔMFCC	94.6	91.5	92.3

PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC +entropy+ Δentropy+ΔMFCC	96	94.4	95.6
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC +entropy+ Δentropy+ΔMFCC	70	70.8	91.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC	50	50.8	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC	50	50	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC	94.4	92.9	94.4
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC +entropy+ Δentropy+ΔMFCC	96	93.8	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC +entropy+ Δentropy+ΔMFCC	90	71.7	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC	51	50.5	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ ΔentropY+ΔMFCC	55	56	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC	95.4	92.1	92.9

14.Step 14: Fusion of 14 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy+ Δentropy+ΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy+ ΔentropY+ΔMFCC (table 14) from step 13 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS (table 15).

Table 15 Fusion of 14 features.

Features	**Speaker identification accuracy**		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS	96.7	92.1	93.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS	77	70.7	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS	51	51.5	91
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS	55	56	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS	95	94.4	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC +entropy+ ΔentropY+ΔMFCC+ΔΔMFCC	77.6	70.7	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔΔMFCC	52	50.5	91.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ ΔentropY+ΔMFCC+ΔΔMFCC	55	56	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔΔMFCC	95.6	93.1	92.1

15.Step 15. Fusion of 15 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS (table 15) from step 14 with the remaining features. Best selected model from this step are

PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS+ΔΔentropy (table 16).

Table 16 Fusion of 15 features.

Features	Speaker identification accuracy		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC	60	70	92.3
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC	67	49.8	91.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC	50	50.8	91.7
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC	95.8	91.9	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroiD+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔentropy	67	71.7	92.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+Δcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔentropy	50	49.7	92.1
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔcentroid+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔentropy	51.5	51.7	92.3

16.Step 16: Fusion of 16 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔENTROPY+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔentropy (table 16) from step 15 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+centroid and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid (table 17).

Table 17 Fusion of 16 features.

Features	Speaker identification accuracy		
	LD	**KNN**	**Ensemble**
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+centroid	46	46.2	92.9
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid	58	53.5	93.5
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+ΔΔcentroid	50	50.6	90.8

17.Step 17: Fusion of 17 features are done by combining best model PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid+centroid and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid (table 17) from step 16 with the remaining features. Best selected model from this step are PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid+centroid and PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+ΔΔcentroid+centroid (table 18).

Table 18 Fusion of 17 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy +ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid +centroid	50	51.2	88.8
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy +ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+ΔΔcentroid +centroid	60	50.6	93.3

18Step 18: Combine all the 18 features. This is the final step of feature fusion.

Table 19 Fusion of 18 features.

Features	Speaker identification accuracy		
	LD	KNN	Ensemble
PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+ entropy+Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC+Δcentroid +centroid+ΔΔcentroid	60	50	90.6

We have ultimately identified the best 35-feature model listed below. We will conduct training and testing on this selected fusion model using ELSDSR, VCTK, and VoxCeleb1 data. This approach is chosen to streamline the process, as training and testing on all 315 models would add unnecessary complexity.

1. MFCC
2. PLP
3. PLP+LPC
4. MFCC+Δentropy
5. PLP+LPC+ΔΔLPC
6. PLP+LPC+ΔPLP
7. PLP+LPC+ΔPLP+MFCC
8. PLP+LPC+ΔPLP+ΔΔLPC
9. PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP
10. PLP+LPC+ΔPLP+ΔΔLPC+RMS
11. PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP+ΔRMS
12. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC
13. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC
14. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS
15. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS
16. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP
17. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔLPC
18. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔLPC
19. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔMFCC+ΔLPC
20. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+entropy+ΔLPC
21. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔMFCC+ΔRMS+ΔΔRMS+ΔLPC+ΔMFCC
22. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ Δentropy+ΔLPC+entropy
23. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔMFCC+ΔLPC+entropy+ Δentropy
24. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy+ Δentropy
25. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC+entropy+ Δentropy+ΔMFCC
26. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy+ Δentropy+ΔMFCC
27. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS
28. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS
29. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropY+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS +ΔΔMFCC
30. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+centroid+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS+Δ Δentropy

31. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS +ΔΔMFCC+centroid
32. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS +ΔΔMFCC+Δcentroid
33. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ΔentropY+ΔMFCC+ΔRMS +ΔΔMFCC+Δcentroid+centroid
34. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS +ΔΔMFCC+ΔΔcentroid+centroid
35. PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy+ΔMFCC+ΔRMS +ΔΔMFCC+Δcentroid+centroid+ΔΔcentroid

For noisy datasets same methodology is applied using TIMIT white noise 630 speaker data and 35 model is selected for the remaining TIMIT white noise data 120 speaker, TIMIT babble noise data 630,120 speaker.
Following is the list of feature model selected for noisy datasets.

1. LPC
2. PLP
3. LPC+MFCC
4. LPC+PLP
5. LPC+PLP+ΔMFCC
6. LPC+PLP+Δentropy
7. LPC+PLP+ΔMFCC+ΔΔLPC
8. LPC+PLP+ΔMFCC+ΔPLP
9. LPC+PLP+ΔMFCC+ΔΔLPC+ΔLPC
10. LPC+PLP+ΔMFCC+ΔPLP+MFCC
11. LPC+PLP+ΔMFCC+ΔΔLPC+ΔLPC+MFCC
12. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy
13. LPC+PLP+ΔMFCC+ΔΔLPC+ΔLPC+MFCC+ΔΔRMS
14. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy
15. LPC+PLP+ΔMFCC+ΔΔLPC+ΔLPC+MFCC+ΔΔRMS+entropy
16. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS
17. LPC+PLP+ΔMFCC+ΔΔLPC+ΔLPC+MFCC+ΔΔRMS+entropy+ΔRMS
18. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+ΔENTRopy+ΔΔRMS+entropy
19. LPC+PLP+ΔMFCC+ΔΔLPC+ΔLPC+MFCC+ΔΔRMS+entropy+ΔRMS+ΔΔentropy
20. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS
21. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC
22. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔΔPLP
23. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔLPC
24. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP
25. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS
26. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔLPC+ΔΔMFCC
27. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔMFC C
28. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔLPC-14
29. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔLPC +ΔΔMFCC
30. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔMFC C+ΔΔLPC
31. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔMFC C+ΔΔLPC+centroid
32. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔMFC C+ΔΔLPC+Δcentroid
33. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔMFC C+ΔΔLPC+centroid+Δcentroid
34. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔENTROPY+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔ MFCC+ΔΔLPC+centroid+ΔΔcentroid
35. LPC+PLP+ΔMFCC+ΔPLP+MFCC+ΔΔentropy+Δentropy+ΔΔRMS+entropy+RMS+ΔLPC+ΔΔPLP+ΔRMS+ΔΔMFC

C+ΔΔLPC+centroid+Δcentroid+ΔΔcentroid

Table 20 presents the total number of models tested when opting for the selection of 1, 2, and 3 best models. In our proposed approach, we have chosen to utilize 2 best models to achieve optimal results within a shorter timeframe, considering that selecting more models would be time-consuming. Figures 9, 10, and 11 illustrate the proposed feature fusion methodology, workflow, and the steps involved in speaker identification and speaker verification computations, respectively.

Table 20 Total number of models testing using TIMIT white noise database (630 speakers).

Step	**Number of feature fusion**	**Total number of models training using 1, 2 and 3 best models at each feature fusion step**		
		1 model	**2 model [Proposed]**	**3 model**
1	1	18	18	18
2	2	17	34	51
3	3	16	32	48
4	4	15	30	45
5	5	14	28	42
6	6	13	26	39
7	7	12	24	36
8	8	11	22	33
9	9	10	20	30
10	10	9	18	27
11	11	8	16	24
12	12	7	14	21
13	13	6	11	18
14	14	5	9	15
15	15	4	7	12
16	16	3	3	9
17	17	2	2	6
18	18	1	1	1
		Total=171	**Total=315**	**Total=475**

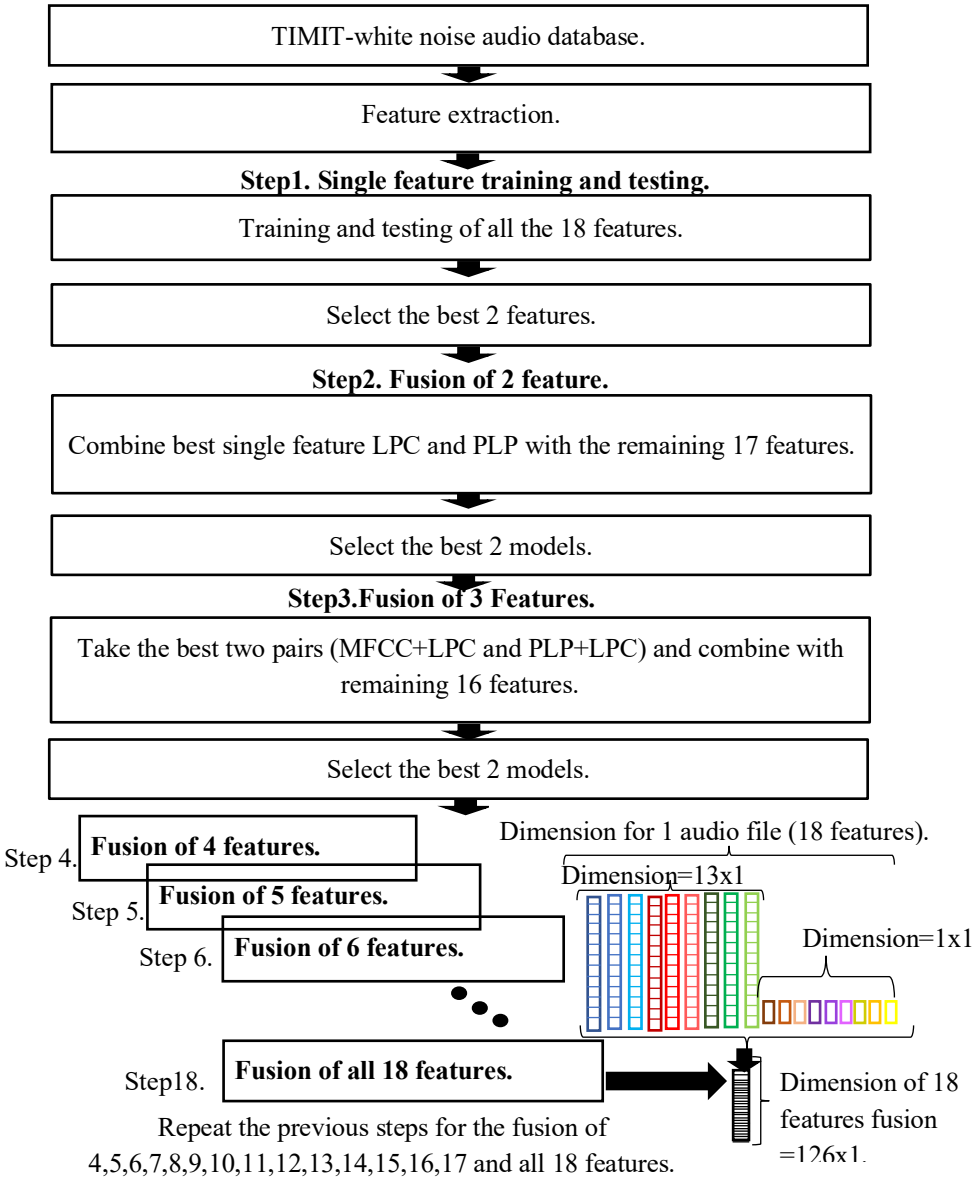


Fig.9 Feature fusion methodology (Approach 1).

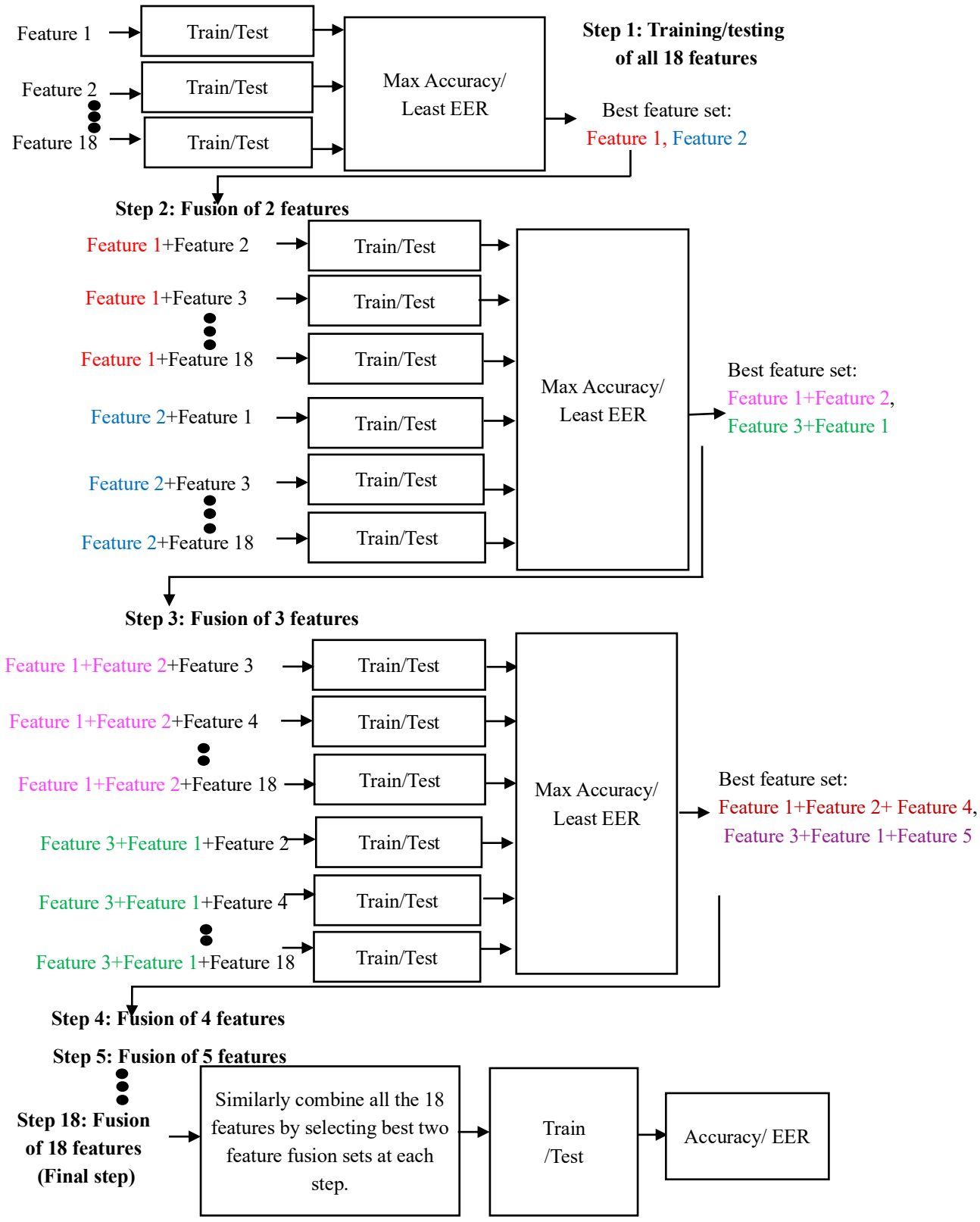


Fig.10 feature level fusion flowchart.

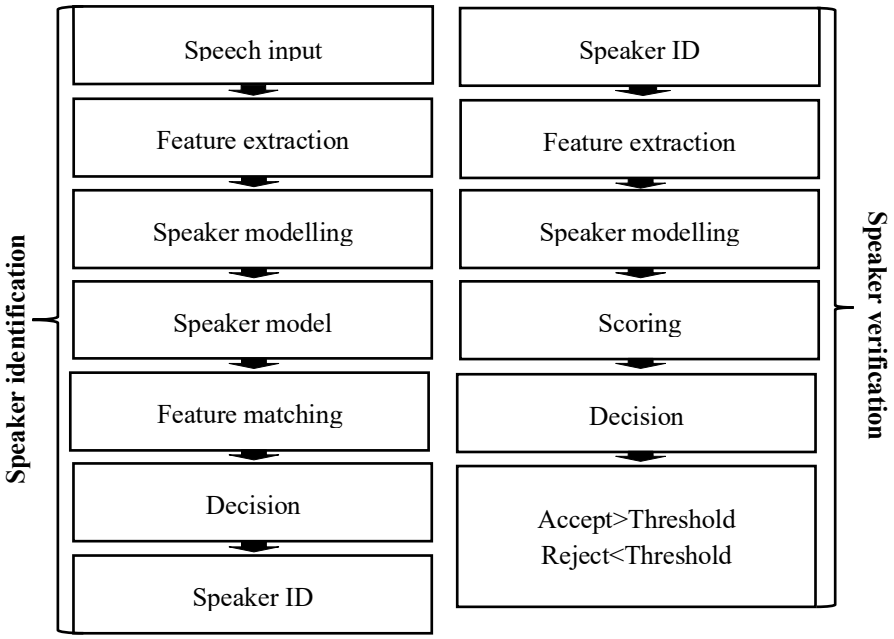


Fig.11 Speaker identification and speaker verification computation steps.

Chapter 3

Methodology for feature dimension reduction and feature pruning

3.1 Dimension reduction techniques

In machine learning and data science, it is essential to identify the most influential features that impact the output results. However, datasets often contain numerous features, leading to overfitting and slow computation processes. To address these challenges, dimension reduction algorithms have been developed to reduce the dimensionality of training data and expedite computations.

Among these techniques, Principal Component Analysis (PCA) is widely used for dimension reduction. It aims to find orthogonal directions, known as principal components, that capture the maximum variance in the data [71]. On the other hand, independent component analysis (ICA) differs from PCA in terms of projection direction. ICA seeks to separate a multivariate signal into additive subcomponents, assuming that the original signals are statistically independent [71].

3.1.1 Principal component analysis (PCA)

Principal component analysis (PCA) is a statistical procedure commonly used for dimensionality reduction. It involves an orthogonal transformation that converts a dataset of correlated variables into a set of linearly uncorrelated variables known as principal components. Matlab, a popular software, provides a built-in PCA function for implementation. The steps involved in PCA are as follows [71-72]:

1. Loading the input data: The feature fusion model, which serves as the input dataset, is loaded into the PCA algorithm.
2. Subtracting the mean: The mean of the data is subtracted from each feature in the original dataset. This step ensures that the data is centroid around the origin.
3. Calculating the covariance matrix: The covariance matrix of the dataset is computed. This matrix captures the relationships and variations among the different features.
4. Determining the eigenvectors: The eigenvectors associated with the largest eigenvalues of the covariance matrix are identified. These eigenvectors represent the directions of maximum variance in the dataset.
5. Projecting the dataset: The original dataset is projected onto the eigenvectors obtained in the previous step. This projection transforms the data into a lower-dimensional subspace spanned by the eigenvectors.

By following these steps, PCA effectively reduces the dimensionality of the dataset while preserving the most important information and capturing the most significant variations in the data [73].

3.1.2 Independent component analysis (ICA)

ICA was first introduced in the 80s by J. Herault, C. Jutten and B. Ans, and the authors proposed an iterative real-time algorithm [64]. independent component analysis (ICA) is a dimensionality reduction technique that aims to extract independent components from a dataset. It is an extension of PCA and provides a way to uncover hidden factors or sources that contribute to the observed data. ICA has gained significant attention in signal processing and data analysis due to its ability to separate mixed signals into their original sources.

The main steps involved in ICA can be summarized as follows:

1. Preprocessing: Similar to PCA, the data is typically preprocessed by centroid and scaling the features to ensure a common reference point and equal contribution of each feature.
2. Whitening: Whitening is performed to transform the data into a new representation where the features are uncorrelated and have unit variances. This step helps to remove any linear dependencies between the features.
3. Defining the non-gaussianity measure: ICA aims to find components that are as statistically independent as possible.

Different non-gaussianity measures can be used, such as kurtosis or negentropy, to quantify the departure from gaussianity and guide the separation of independent components.

4. Optimization: The main objective of ICA is to maximize the non-gaussianity measure for each component. This is achieved through an optimization process, which involves finding the weights or mixing matrix that maximizes the non-gaussianity measure.
5. Iterative estimation: ICA often involves an iterative estimation process to refine the separation of independent components.
6. Reconstruction: Once the independent components are obtained, they can be used to reconstruct the original dataset or further analyzed for specific purposes such as feature extraction or signal separation.

The specific algorithmic details and mathematical models of ICA can be found in the referenced papers [74] and [75]. Fig.12 and fig.13 shows projection suing PCA and ICA dimension reduction techniques.

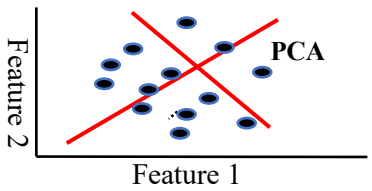


Fig.12 PCA projection.

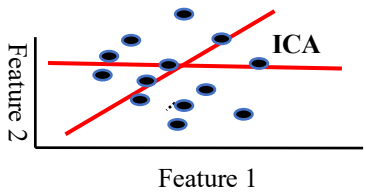


Fig.13 ICA projection.

3.2 Programming

3.2.1 Matlab code for principal component analysis (PCA)

```
%PCA
[coeff, Data_PCA, latent, tsquared, explained, mu] = pca(Data2, 'NumComponents', q);
```

3.2.1.1 Code explanation for PCA

1. **coeff`** contains the principal component coefficients, which represent the directions of the principal components.
2. **Data_PCA** stores the data transformed into the principal component space, allowing for reduced dimensionality representation.
3. **latent** represents the eigenvalues of the covariance matrix of `Data2`, providing information about the variances of the principal components.
4. **tsquared** contains hotelling's T-squared statistic for each observation in `Data2`, which is used to identify multivariate outliers.
5. **explained** holds the percentage of the total variance explained by each principal component, providing insights into the significance of each component.
6. **mu** represents the sample mean of `Data2`, providing information about the centroid tendency of the data.
7. **q** is number of coefficients.

3.2.2 Matlab code for independent component analysis (ICA)

```
Mdl = rica(Data, q);
Data_ICA = transform(Mdl, Data);
```

3.2.2.1 Code explanation for ICA

1. **Mdl** represents the ICA model obtained from the data, encapsulating the extracted independent components and their properties.
2. **Data_ICA** stores the data transformed using the ICA model, resulting in a representation where the extracted components are statistically independent from each other.

3.3 Model optimization using dimension reduction technique.

PCA and ICA, play a crucial role in improving the performance of speaker recognition systems. Following are the steps involve for model optimization using PCA and ICA.

- In our approach, we started by transforming the original 126-feature fusion (18 features) into 126 PCA and 126 ICA features vectors.
- Then, we randomly reduced the dimensions of all 18 features fusion model from 50% to 90% of their original size using PCA and ICA.
- Now, we have 3 new reduced PCA and ICA features model and one 126 PCA and ICA features model.
- Train and test new PCA and ICA models and find the one giving best results.
- To evaluate the performance of the reduced dimension models, we employed LD, KNN, and ensemble classifiers.
- Accuracy, equal error rate (EER), and computation timing were calculated for each reduced model.
- By comparing the results obtained with and without dimension reduction techniques, we were able to assess the impact of dimensionality reduction on the speaker recognition performance.

The key goal of our study was to identify the best model that achieves optimal SR performance while effectively reducing the dimensionality of the feature space. Through the comparison and analysis of the different reduced dimension models generated using PCA and ICA, we were able to identify the model with the highest accuracy and lowest EER. This model represented the optimal balance between dimensionality reduction and speaker recognition performance. Fig.14 explains steps involved in model optimization using dimension reduction technique for each dataset used.

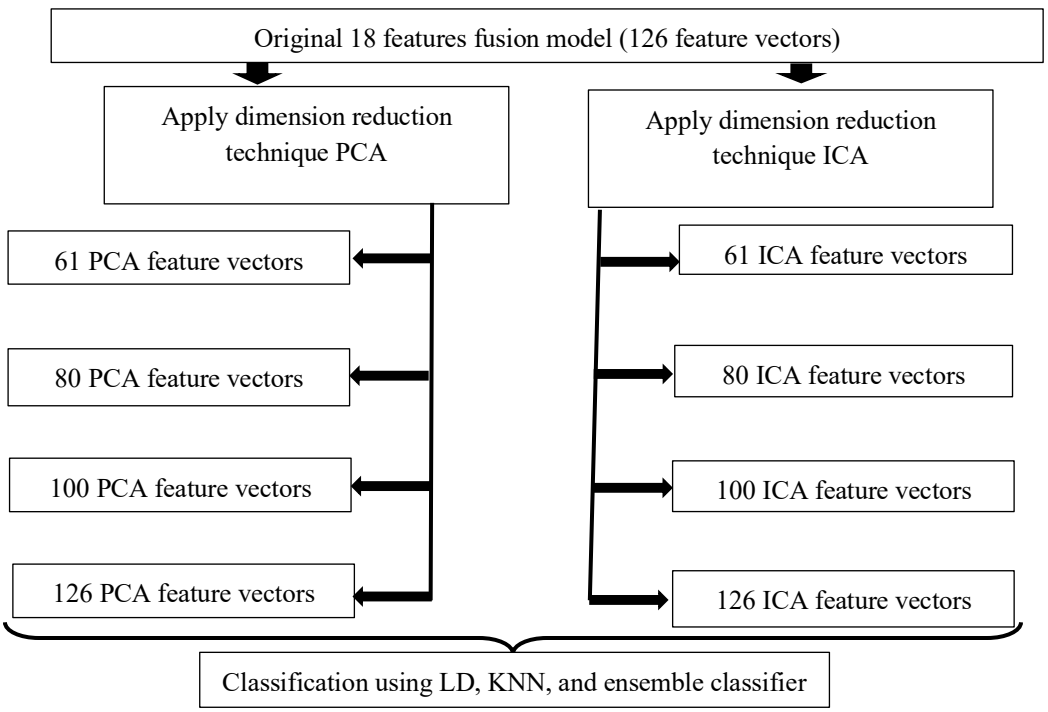


Fig.14 Model optimization using dimension reduction technique (Approach 2).

3.4 Feature optimization model

Feature selection is a crucial step in machine learning and data analysis tasks as it aims to identify the most relevant and informative features that contribute significantly to the output results. The main purpose of feature selection is to enhance model performance, reduce computational complexity, and mitigate the risk of overfitting. While PCA and ICA are effective dimensionality reduction techniques, they do not inherently provide feature selection capabilities. PCA and ICA focus on transforming the original feature space into a new set of uncorrelated variables or independent components, respectively. However, they do not directly assist in identifying the most important features or determining the optimal number of features to retain.

In contrast, feature selection approaches address this limitation by systematically evaluating the relevance and contribution of each feature in the dataset. By employing feature selection techniques, the effort and time required for manual feature selection are reduced, as the algorithm automatically identifies the most informative features. This automated approach helps streamline the feature selection process and improves the efficiency of speaker recognition or other applications. [4-8].

Proposed work has used wrapper-based feature selection with KNN classifier. Wrapper-based feature selection offers optimality for specific algorithms, considers feature interactions, and provides flexible and adaptive selection, leading to more accurate and context-specific feature subsets. The referenced paper [76] may provide insights into accelerating this process using the K-nearest neighbor (KNN) algorithm, which can help reduce the computational burden while still leveraging the advantages of wrapper-based feature selection.

In this proposed work we have used genetic algorithm (GA) and marine predators algorithm (MPA) feature selection method. Following point explains the parameters and importance of GA and MPA feature optimization method.

3.4.1 Genetic algorithm (GA) [6],[7]

Genetic Algorithms (GAs) function by iteratively refining a pool of potential solutions. This process involves employing methods such as selection, crossover (mixing), and mutation, with the objective of producing enhanced generations of solutions. The effectiveness of genetic algorithms lies in their ability to explore extensively, manage complexity, and adapt to dynamic changes, making them proficient in identifying optimal features [6],[7]. Following figure 15 shows algorithm step for GA.

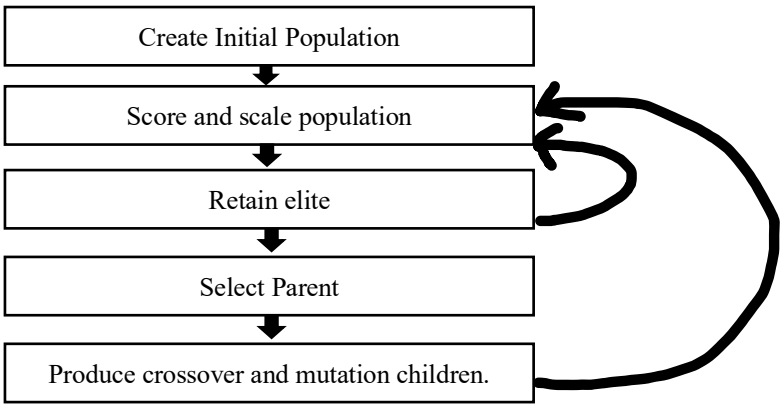


Fig.15 Flowchart for GA.

3.4.1.1 GA optimization steps

The feature selection process using the genetic algorithm (GA) with a maximum of 100 iterations with KNN classifier is used in the proposed work genetic algorithms (GAs) and k-nearest neighbors (KNN) are two distinct techniques used in machine learning, but they can be combined to enhance certain aspects of the learning process. Combining genetic algorithms with k-nearest neighbors can bring several benefits, such as improved feature selection, hyperparameter tuning, distance metric learning, handling imbalanced datasets, optimization of ensemble methods, and better scalability for large datasets.

Following parameters and steps are important when apply GA for feature selection.

1. **Set the following parameters used:**

- opts.N: Number of solutions in the population (e.g., opts.N = 10).
- opts.T: Maximum number of iterations (e.g., opts.T = 100).
- CR: Crossover rate (e.g., CR = 0.8).
- MR: Mutation rate (e.g., MR = 0.01).

2. **Initialize the population:** Generate an initial population of opts.N candidate solutions (K-feature subsets) randomly.
3. **Evaluate the fitness of each candidate solution.** For each candidate solution:

- Reduce the acoustic vectors of the entire speech database to the features enumerated in the candidate solution.
- Estimate speaker models using a training corpus.
- Classify utterances in a development corpus using the speaker models.
- Calculate the speaker recognition accuracy obtained for the development corpus as the fitness score for the candidate solution.

4. **Genetic operations:**

- Perform the following steps until the maximum number of iterations (opts.T) is reached:
- Select the fittest candidate solutions based on their fitness scores.
- Apply crossover to create new candidate solutions.
- Randomly select two parent solutions from the fittest solutions.
- Generate two offspring solutions by exchanging genetic information (features) between the parents based on the crossover rate (CR).
- Apply mutation to introduce small variations in the offspring solutions.
- For each offspring solution, randomly mutate individual features with a probability of the mutation rate (MR).
- Evaluate the fitness of the new candidate solutions.
- Replace a portion of the population with the new candidate solutions, including the fittest individuals (elitism).
- Repeat the above steps for the specified number of iterations.

5. **Termination:**

- Check if the maximum number of iterations (opts.T) is reached or if the fitness improvement is smaller than a given threshold.
- If the termination condition is met, stop the algorithm and return the optimal K-feature subset (Γ) that achieved the highest fitness score.

6. **Evaluation:** Evaluate the optimal K-feature subset (Γ) on an independent test corpus to assess its performance [6],[7].

3.4.1.2 Matlab code for genetic algorithm (GA)

```
clear, clc, close.
% Number of k in K-nearest neighbor
opts.k = 1;
% Ratio of validation data
ho = 0.4;
% Common parameter settings
CR = 0.8;    % crossover rate
MR = 0.01;   % mutation rate
% Parameter of MPA
opts.b = 1;
% Load dataset
load DATA.mat;
% Divide data into training and validation sets
HO = cvpartition(label,'HoldOut',ho);
opts.Model = HO;
% Perform feature selection
FS     = jfs('GA',feat,label,opts);
% Define index of selected features
sf_idx = FS.sf;
% Accuracy
Acc    = jknn(feat(:,sf_idx),label,opts);
```

3.4.1.3 Code explanation

This matlab code appears to perform feature selection and classification using the K-nearest neighbors (KNN) algorithm. Here's a breakdown of the code:

1. **clear, clc, close:** These commands clear the workspace, command window, and any open figures to start with a clean environment.
2. **opts.k = 1:** Sets the value of `k` to 1, which is the number of neighbors used in the K-nearest neighbors algorithm.
3. ho = 0.4;`: Defines the hold-out ratio, where 40% of the data will be used for validation, and the remaining 60% will be used for training.
4. **CR = 0.8: and MR = 0.01:** These parameters represent the crossover rate (CR) and mutation rate (MR), which are typically used in genetic algorithms for feature selection and optimization.
5. **opts.b = 1:** Sets a parameter `b` to 1, which appears to be a specific parameter used in the feature selection algorithm (MPA).
6. **load DATA.mat:** Loads a dataset from a file named 'DATA.mat'. This dataset likely includes features (feat) and corresponding labels (label).
7. **HO = cvpartition(label,'HoldOut', ho):** Creates a hold-out cross-validation partition based on the 'label' data, where 40% of the data is held out for validation (as specified by `ho`). This will be used to split the dataset into training and validation sets.
8. **opts.Model = HO:** Assigns the created hold-out cross-validation partition (`HO`) to the 'Model' parameter in the options (`opts`).
9. **FS = jfs('GA', feat, label, opts):** This line performs feature selection using a genetic algorithm (GA) specified by 'GA'. It selects the most relevant features from the dataset (`feat`) based on the labels (`label`) and the defined options (`opts`).
10. **sf_idx = FS.sf:** Extracts the indices of the selected features from the feature selection results and stores them in `sf_idx`.
11. **Acc = jknn(feat(:,sf_idx),label,opts):** Uses the selected features (`feat(:,sf_idx)`) to perform k-nearest neighbor (KNN) classification on the dataset, and calculates the classification accuracy (`Acc`). The options (`opts`) used in the KNN algorithm are also specified.

3.4.2 Marine predator's algorithm (MPA)

The marine predator algorithm (MPA) is like a computer program inspired by the way animals in the ocean, like predators (hunters) and prey (those they hunt), interact. It helps the computer choose the best features for recognizing different speakers when they talk. This is similar to how predators in the sea capture prey to survive. It's useful for improving tasks like voice recognition [8],[9]. Optimization process in MPA comprises three distinct phases, illustrated in the figure. These phases are categorized based on the velocity ratio and time:

- 1. Phase 1: The predator moves at a slower pace than the prey, characterized by a high velocity ratio.
- 2. Phase 2: The predator and prey maintain nearly identical speeds, representing a unity velocity ratio.
- 3. Phase 3: The predator accelerates and moves faster than the prey, indicating a low velocity ratio.

Following figure 16 shows the flow for the MPA algorithm.

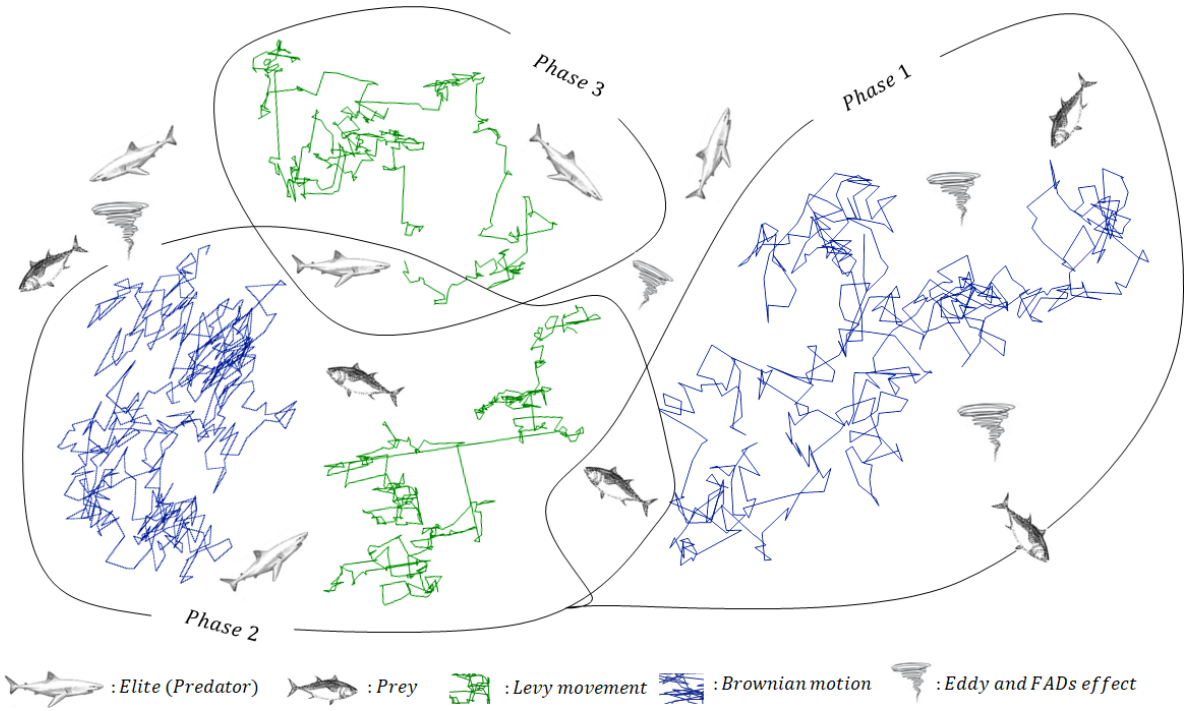


Fig.16 MPA algorithm.
(source: https://www.mathworks.com/matlabcentral/fileexchange/74578-marine-predators-algorithm-mpa)

3.4.2.1 Feature optimization using MPA

The provided code snippet represents a function called jmarine predators algorithm that implements the marine predators algorithm (MPA) for feature selection in combination with the K-nearest neighbors (KNN) classifier. The function takes three input arguments: `feat` (the feature matrix), `label` (the corresponding labels), and `opts` (a structure containing algorithm parameters).
The parameters used in the MPA implementation are as follows:

- lb and ub: The lower and upper bounds for the position values of the marine predators.
- thres: A threshold value used for determining the movement of the marine predators.
- beta: A parameter representing the levy component, which contributes to the random movement of the marine predators.
- P: A constant value used in the algorithm.
- FADs: The fish aggregating devices effect, representing the influence of the prey distribution on the movement of the marine predators.

To understand the complete steps and implementation details of the marine predators algorithm with the KNN classifier using the provided parameters, it is recommended to refer to the referenced papers [8],[9].

3.4.2.2 Matlab code for marine predator algorithm (MPA)

```
clear, clc, close;
% Number of k in K-nearest neighbor
opts.k = 1;
% Ratio of validation data
ho = 0.4;
% Common parameter settings
opts.N  = 10;     % number of solutions
opts.T  = 100;    % maximum number of iterations
% Parameters of MPA
```

```
lb = 0;
ub = 1;
thres = 0.5;
beta = 1.5;   % levy component
P = 0.5;   % constant
FADs = 0.2;   % fish aggregating devices effect
% Load dataset
load DATA.mat;
% Divide data into training and validation sets
HO = cvpartition(label,'HoldOut',ho);
opts.Model = HO;
% Perform feature selection
FS = jfs('mpa',feat,label,opts);
% Define index of selected features
sf_idx = FS.sf;
% Accuracy
Acc = jknn(feat(:,sf_idx),label,opts);
```

3.4.2.3 Code explanation

1.**opts.k = 1:** Sets the value of the number of nearest neighbors (k) for the K-nearest neighbors algorithm to 1.
2.**ho = 0.4:** Defines the ratio of data to be used for validation as 40%.
3 **opts.N = 10 and opts.T = 100:** Sets the number of solutions and the maximum number of iterations for the optimization process.
4.Parameters for the marine predator algorithm (MPA) are specified, including the lower bound (`lb`), upper bound (`ub`), threshold value (`thres`), beta value for the Levy component (`beta`), a constant (`P`), and the fish aggregating devices effect (`FADs`).
5.**load DATA.mat:** Loads a dataset from a file named 'DATA.mat'.
6.**HO = cvpartition(label,'HoldOut',ho):** Divides the data into training and validation sets using a hold-out cross-validation partition.
7.Feature selection is performed using the Marine Predator Algorithm (MPA) through the `jfs` function with the specified parameters and options.
8.The indices of the selected features are stored in the variable `sf_idx`.
9.The code uses the selected features to perform K-Nearest Neighbors (KNN) classification on the dataset.
10.The accuracy of the classification is computed and stored in the variable `Acc`.
This code effectively combines the marine predator algorithm (MPA) for feature selection and the K-nearest neighbors (KNN) algorithm for classification to identify and use the most relevant features for accurate classification.

3.5 Model optimization using feature selection approach.

In the feature selection process, three sets of feature vectors were considered: the original feature vectors, the feature vectors obtained from principal component analysis (PCA), and the feature vectors obtained from independent component analysis (ICA). By considering these three sets of feature vectors (original, PCA, and ICA), the feature selection approaches genetic algorithms (GA) and marine predators algorithm (MPA) were applied.

After applying GA and MPA to the original 18 feature, PCA and ICA 126 features we get PCA-GA, PCA-MPA, ICA-GA, ICA-MPA, features-GA, features-MPA features vectors (fig.8) and accuracy EER value and computation time is calculated and compared with other approaches using all three classifier used. The goal was to find the best combination of features that would optimize these performance measures for the specific task at hand.

Comparing the results obtained from feature selection using GA and MPA for the original, PCA, and ICA feature vectors allowed for the identification of the most effective approach in terms of classification accuracy, EER, and computation time for each set of feature vectors. Following figure 17 explains proposed feature optimization steps (approach 3).

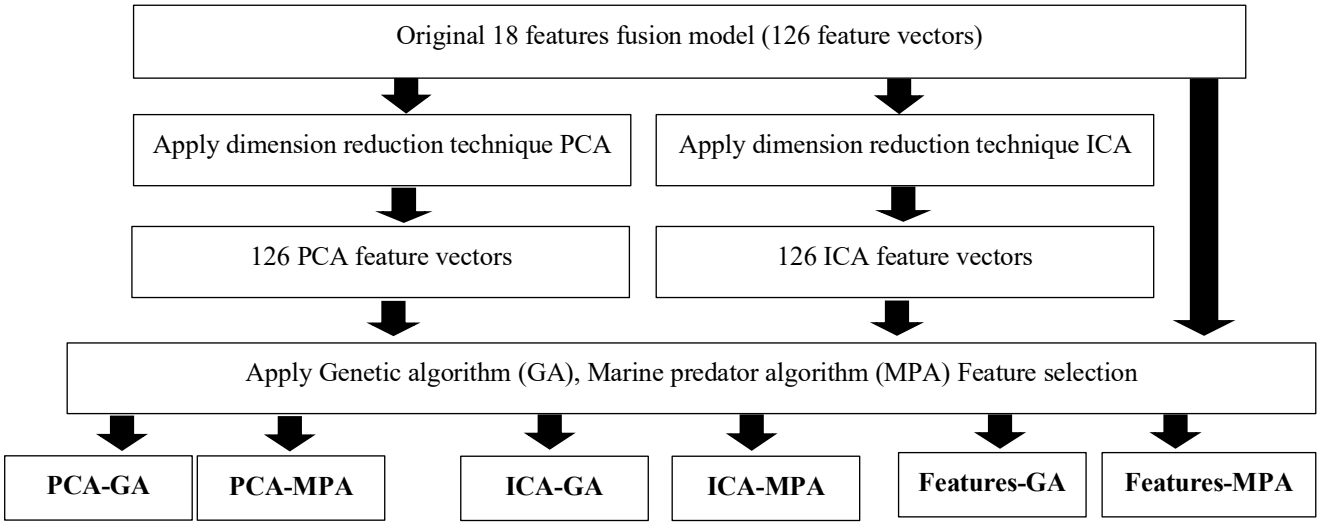


Fig.17 Model optimization using feature selection methods.

Chapter 4

Evaluation and performance analysis of speaker recognition systems

4.1 Classification methods used for speaker recognition system

4.1.1 Linear discriminant classifier (LD)

LD (linear discriminant) uses Bayes' theorem to calculate the probabilities. Given the output class as (k) and the input as (x), Bayes' theorem is applied to determine the probability that the data belongs to each class, as shown in equations 18 and 19.

$$P(Y=x|X=x) = (PIk * fk(x)) / sum(PII * fl(x)) \quad (18)$$

$$PIk = nk/n \quad (19)$$

In the above equation, PIk represents the prior probability, which is the base probability of each class as observed in the training data. The function f(x) is an estimated probability that x belongs to a particular class and employs a gaussian distribution function. Here, n denotes the number of instances, and K is the number of classes. By substituting the gaussian distribution into the equation and simplifying, we arrive at equation 20. This function is a discriminant, and the class with the highest calculated value will be the output classification (y).

$$Dk(x) = x * (muk/siga^2) –(muk^2/(2*sigma^2)) + ln(PIk) \quad (20)$$

Dk(x) represents the discriminant function for class k given input x, where muk, sigma^2, and PIk are all calculated from the data. For a comprehensive understanding of LD classification, detailed explanations can be found in reference [77]. Fig. 18 shows the class division using LDA.

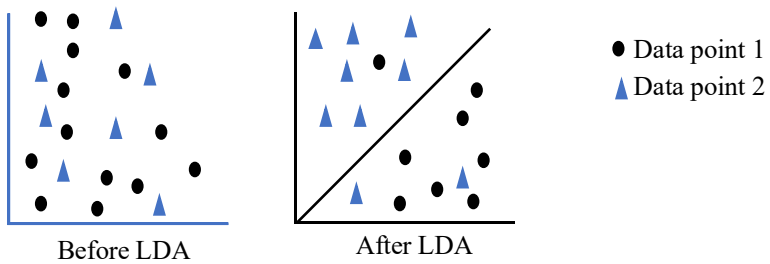


Fig.18 Linear discriminant classification.

4.1.2 K Nearest Neighbor classification (KNN)

The K-nearest neighbors (KNN) approach is a classification method used to categorize unknown data points based on their similarity to neighboring data points (figure 19). In this approach, the parameter K determines the number of dataset elements that contribute to the classification process, and for this specific work, K is set to 1. The KNN algorithm can be summarized in the following steps.

1. Select a value for K, representing the number of neighbors to be considered during classification.
2. Calculate the euclidean distance between the unknown data point and its K nearest neighbors.
3. Identify the K nearest neighbors based on the computed euclidean distances.
4. Count the number of data points in each class among these K neighbors.

5. Assign the new data point to the class with the highest count among the K neighbors. For a more in-depth understanding of the KNN algorithm, please refer to reference [78].

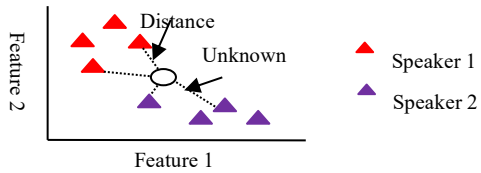


Fig.19 K nearest neighbour (KNN) .

4.1.3 Ensemble classification

In this work, an ensemble classification method is employed to enhance the speaker recognition (SR) results and mitigate overfitting [79]. The proposed ensemble classifier utilizes the random subspace ensemble method, comprising 30 learners with a subspace dimension of 5. The random subspace ensemble approach, also known as bagging or feature bagging, combines predictions from multiple decision trees trained on various subsets of columns in the training dataset. This method reduces the correlation between the estimators by training them on random samples of features rather than using the entire feature set [80].

To train the ensemble classification, the Fitcensemble function in MATLAB is utilized. The data matrix X contains observations, where each row represents a single observation, and each column represents a predictor variable. The response vector Y has the same number of observations as the rows in X. To calculate an ensemble of models using the random subspace method, the following algorithm is employed:

1. Assume N is the number of training points, and D is the number of features in the training data. L represents the number of individual models in the ensemble.
2. For each individual model L, select nl (where nl < N) as the number of input points. It is typically the same value for all individual models.
3. For each individual model L, generate a training set by randomly selecting dl features from D (with replacement) and train the model.
4. When applying the ensemble model to a hidden point, combine the outputs of the L individual models using majority voting [80].

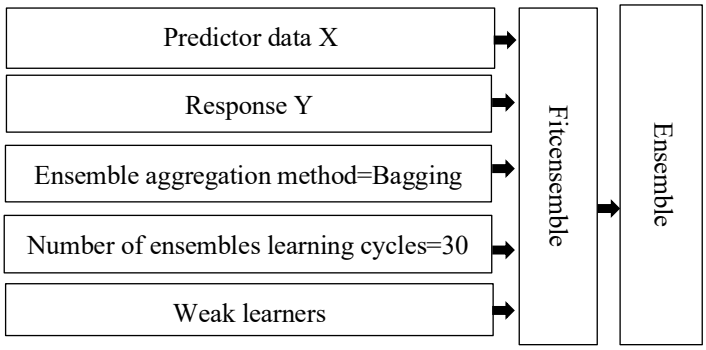


Fig.20 Ensemble classification.

4.1.4 Classification learner app

For the accuracy and EER calculation we have used the classification learner app

The classification learner app in matlab facilitates the training of models for data classification. This tool allows you to delve into supervised machine learning through the utilization of various classifiers. With this app, you can:

1. Explore your dataset thoroughly.
2. Select features relevant to your classification task.
3. Define and implement appropriate validation schemes.
4. Train models using different algorithm K nearest neighbour,ensemble,Linear discriminant classification.

5. Evaluate and analyze the performance of trained models.

By feeding the app with labeled input data, you can train a model that can make predictions for new, unseen data. Moreover, you have the option to export the trained model to the workspace or generate matlab code for future use or programmatic modifications, enabling you to customize the classification process as needed.

Figure 21 shows the accuracy results using classification learner application.

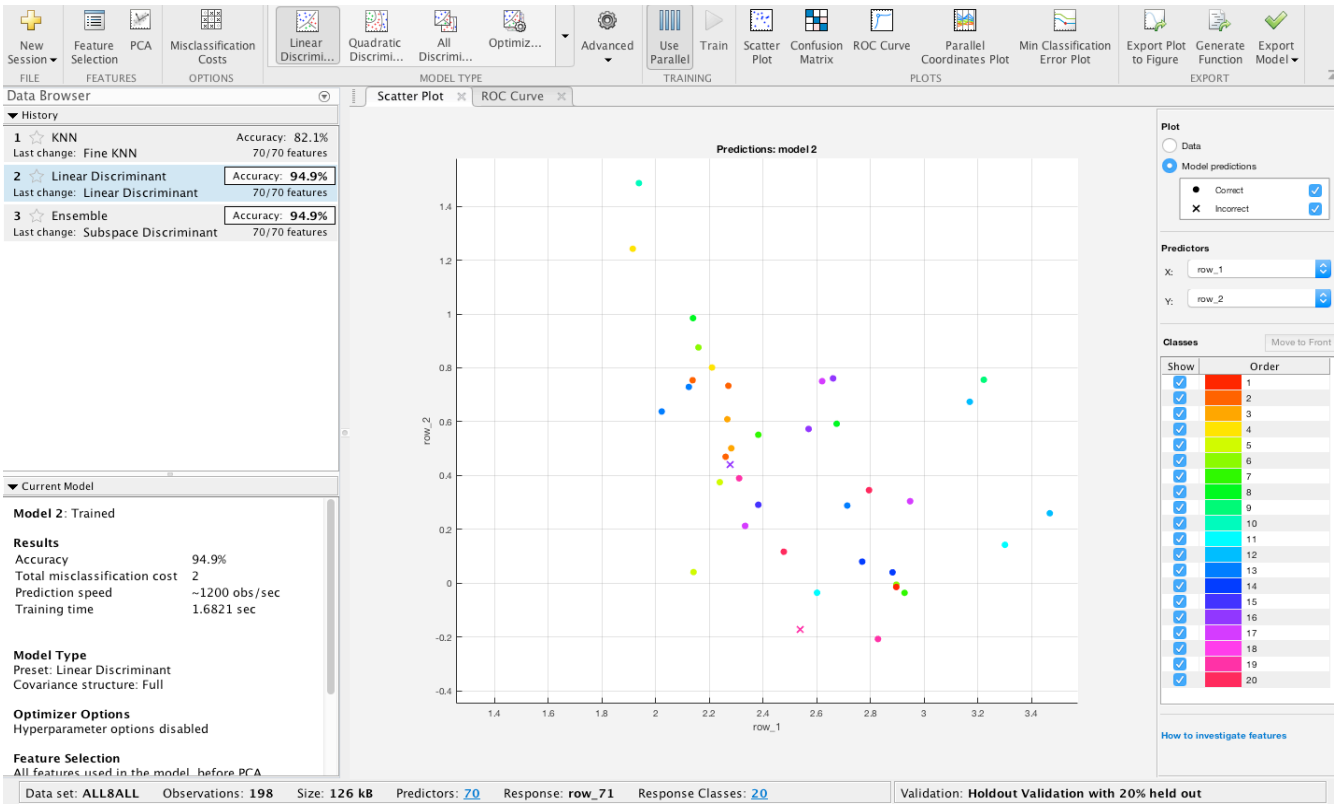


Fig.21 Screenshot for accuracy calculation using classification leaner app.

Note: To accurately calculate the classification performance using the classification learner app, different colors can be used to represent data points for different speakers. This color-coded representation aids in visualizing and analyzing the performance of the classification model. By assigning unique colors to different speakers, the classification of the data points becomes more apparent. This allows for a better understanding of how well the model is distinguishing between the different speakers.

4.2 Database preparation

The following seven voice databases are used in the proposed work, and their explanations are as follows.

We conducted comprehensive tests on both clean and noisy voice data to ensure the effectiveness of our methodology across various types and sizes of voice data. The primary objective of this research is to achieve optimal performance across diverse forms of voice data.

1. ELSDSR is a small corpus dataset that was recorded at the technical university of Denmark (DTU) by faculty, Ph.D. students, and master's students. ELSDSR consists of voice messages from 22 speakers, 12 males and 10 females. For training, 154 voices were recorded with 7 sentences each. For the testing set, 44 utterances were provided, and 2 sentences were spoken by each speaker. The time duration for the training data is 78 seconds for males and 88.3 seconds for females. The test data duration is 16.1 seconds for males and 19.6 seconds for females [81].
2. Voxforge is an open speech dataset (medium size) consisting of many speaker voices. For the proposed work, 100 English speakers are randomly selected. Each speaker spoke 10 sentences recorded at a sampling rate of 8 kHz. A total of 1000 voice files are used for 100 speakers. Out of the 1000 voices, 800 voices are used for training, and 200 voices are used for testing [49].

3. The CSTR VCTK (medium size) corpus consists of speech data from 109 native English speakers with different accents. Each speaker recorded approximately 400 English sentences. For the proposed work, 5 sentences from each speaker are selected. A total of 545 voices are used. A total of 436 voices are used for training, and the remaining 109 voices are used for testing [49].
4. A total of 942 hours of multilingual telephone speech and English interview speech are included in the NIST-SRE-2008 database (medium size) [54, 82]. The sampling frequency was converted from the original 8 kHz to 16 kHz, and 120 English-only microphone channels were selected for better comparison with other databases. Audacity software [83] is used to separate a single speaker from multiple speakers and segment the speaker voice into 10 equal parts. Each speaker consists of 10 audio files with a fixed length of 8 seconds each. Six audio files are used for training, and the remaining 4 audio files are used for testing. A total of 1200 voices are used, out of which 720 voices are used for training and 480 voices are used for testing.
5. Voxceleb1 (large size) contains more than 100,000 voice samples. Videos included in the database are recorded in challenging multispeaker environments, including at red carpet events and in outdoor stadiums. All the datasets are degraded with real-world noise, such as laughter, overlapping speech and room acoustics. For the proposed work, all 1251 pieces of speaker data are used for a total of 153516 speaker voices. To obtain a fair comparison with [55] and [56], 148642 utterances are used for training and 4874 utterances are used for testing in the SV task; in addition, 145265 utterances are used for training, and 8251 utterances are used for testing. Table 3 provides the details of all the voice datasets used.
6. In this research work, we incorporated the noisy TIMIT speech dataset, developed by the Florida institute of technology, which consists of approximately 322 hours of speech from the TIMIT acoustic-phonetic continuous speech corpus (LDC93S1). The dataset was modified by adding different levels of additive noise while keeping the original TIMIT arrangement intact. For our study, we specifically focused on TIMIT white noise and babble noise with 30dB noise level. We selected subsets of the dataset containing 120 speakers for TIMIT babble and white noise, and 630 speakers for TIMIT white and babble noise. Each speaker contributed a total of 10 utterances. For TIMIT babble and white noise with 120 speakers, we used 720 voice samples for training and 480 voice samples for testing, resulting in a total of 1200 voices.
7. Similarly, for TIMIT babble and white noise with 630 speakers, we used 5040 voice samples for training and 1260 voice samples for testing, totaling 6300 voices [84]. This approach allowed us to make fair comparisons with other studies, including [57] and [58]. The training and testing were specifically conducted for the 30dB babble noise and white noise TIMIT data to ensure a meaningful comparison with previous research and to evaluate the effectiveness of our proposed approach on noisy speech datasets. Table 21 and table 22 shows the details of clean and noisy voice dataset used in the propose work

Table 21 Clean database details.

Information	ELSDSR (Small dataset)	Voxforge (Medium dataset)	VCTK (Medium dataset)	NIST-2008 (Medium dataset)	Voxceleb1 for SI (Large dataset)	Voxceleb1 for SV (Large dataset)
Total number of speakers	22	100	109	120	1251	1251
Each speaker utterance	9	10	5	10	Undefined	Undefined
Total utterance for training	154	800	436	720	145265	148642
Total utterance for testing	44	200	109	480	8251	4874
Total number of audio recordings	198	1000	545	1200	153516	153516
Source	Open	Open	Open	Linguistic data consortium	Open	Open
Language	English	English	English	English	English	English
Environment	Clean	Clean	Clean	Clean	Multimedia	Multimedia

Table 22 Noisy database details.

Information	Voxceleb1 for SI (Large size data)	Voxceleb1 for SV (Large size data)	TIMIT-babble Noise-30DB	TIMIT-babble Noise-30DB	TIMIT-white Noise-30DB	TIMIT-white Noise-30DB
Total number of speakers	1251	1251	630	120	630	120
Each speaker utterance	Undefined	Undefined	10	10	10	10
Total utterance for training	145265	148642	5040	720	5040	720
Total utterance for testing	8251	4874	1260	480	1260	480
Total number of audio recordings	153516	53516	6300	1200	6300	1200
Source	Open	Open	Linguistic data consortium	Linguistic data consortium	Linguistic data consortium	Linguistic data consortium
Language	English	English	English	English	English	English
Environment	Multimedia	Multimedia	Noisy	Noisy	Noisy	Noisy

4.3 Performance evaluation for speaker identification

The SI accuracy (equation 21) is calculated using a classification learner application on MATLAB for all proposed models using all 3 classifiers. To make it simpler only models giving highest accuracy are shown in the result tables. Our primary evaluation criterion is to achieve the highest level of accuracy.

$$Accuracy = \frac{Number\ of\ voices\ correctly\ identified}{Total\ number\ of\ audio\ files} \quad (21)$$

4.4 Performance evaluation for speaker verification

Receiver operating characteristic (ROC) curves, constructed by plotting false positive rate (FPR) and true positive rate (TPR) at 0.005 intervals for each speaker, provide a detailed evaluation of different models. The equal error rate (EER), calculated at the ROC curve's intersection with the diagonal axis from (0,1) to (1,0), represents the false positive rate. An optimal ROC curve, exemplified by the fig. 22 blue line (best models), depicts perfect accuracy with 0 false positives and 0 false negatives. Similar ROC curves are generated by the best-performing model, with curves positioned towards the left corner indicating superior results [85,86]. For each speaker, one is considered the true speaker, while others collectively serve as impostors. Fig. 22 illustrates the schematic for ROC curve construction and EER calculation, and the final EER values are presented in result tables. Classifier performance is comprehensively represented by following outcomes:

- True Positive (TP): Correctly predicting the positive class.
- True Negative (TN): Correctly predicting the negative class.
- False Positive (FP): Incorrectly predicting the positive class.
- False Negative (FN): Incorrectly predicting the negative class.
- Sensitivity (True Positive Rate or Recall) is the ratio of correctly classified positive samples to the total positive samples:

$$TPR = \frac{TP}{TP+FN} \quad (22)$$

- Specificity (True Negative Rate or Inverse Recall) is the ratio of correctly classified negative samples to the total negative samples: $TNR = \frac{TN}{TN+FP}$ (23)
- False Positive Rate (FPR), also known as false acceptance rate (FAR) or fallout, represents the ratio of incorrectly classified negative samples to the total negative samples: $FPR = \frac{FP}{FP+TN}$ (24)
- Additionally, the false negative rate (FNR) or false rejection rate signifies the proportion of positive samples that were incorrectly classified: $FNR = \frac{FN}{FN+TP}$ (25)

This comprehensive explanation enhances clarity, providing a deeper understanding of ROC curves, EER, and the nuanced evaluation of classifier performance through various metrics.

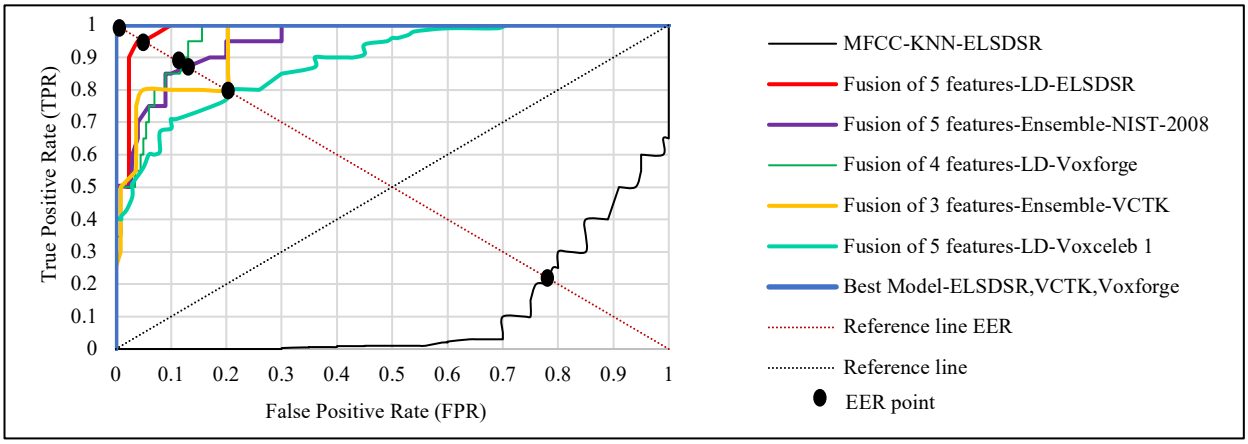


Fig.22 EER calculation using ROC curve for various feature fusion model.

4.5 Result observation

4.5.1 Speaker recognition result using feature level fusion for clean voice dataset.

Tables 23-table 27 show the best model results for clean voice data ELSDSR, voxforge, NIST-2008, VCTK and voxceleb1 databases using feature level fusion approach1 which is first part of our research. Tables 23-27 provide the top SR performance (with the highest SI accuracy and lowest SV EER values) compared to other fusion models, and the results generated by other models. For a fair comparison, we included only the results of the other models that use the same database and number of audio clips as used by the proposed model. Additionally, Tables 23-27 show a comparison of the proposed best model results (red font), the common effective model results on all datasets (bold red font) and other models' best results on the ELSDSR, voxforge, NIST-2008, VCTK and voxceleb1 databases, respectively. The following points explain the number of best models (those that obtain the highest SI accuracy and lowest SV EER values) obtained by the proposed work and one effective model suitable for all the datasets.

1. For the ELSDSR database, the best average SV EER of 0% and SI accuracy of 100% are achieved by the proposed work using a fusion of 6, 7, 9, 10, 12, 14, and 15 features; however, for ELSDSR data, less research has been performed for SV; hence, previous results are difficult to compare. The highest SI accuracies of 98% [43], 95% [44], and 94.8% [48] are obtained with score-level fusion and GMM-UBM modeling, respectively while least EER of 2% is achieved when [43] uses RF-SVM score level fusion. As ELSDSR is a small voice dataset, the proposed models with the fusion of 6, 7, 9, 10, 12, 14 and 15 features can be considered suitable for small audio databases (Table 23).
2. For the voxforge speech database, the proposed models with a fusion of 7 and 14 features obtain the lowest EER value of 0% and the highest SI accuracy of 100% and 99.5% with the KNN classifier, while previous SR models on voxforge speech data [49] achieved an EER value of 0% using the feature-level fusion method when the same number of voices were used for training and testing as used by the proposed model. The best SI accuracies of 94% and 93% are achieved by [51] and [52] using feature fusion and model fusion, respectively (Table 24).

3. For the NIST-2008 database, the proposed models achieve the best EER of 0.2% and the best SI accuracy of 96.9% using a fusion of 11, 12, and 14 features with an LD classifier, and the score fusion method with GMM-UBM modeling [53] achieves the best accuracy of 95.83%. The model in [54] achieved the best SI accuracy of 96.6% with an i-vector approach, and with GMM-UBM modeling, a best accuracy of 95.83% is achieved when tested on the same NIST-2008 data. [53] and [54] included only SI results; therefore, only the SI results of proposed work can be compared with the results in [53] and [54] (Table 25).
4. For the proposed models using 5, 8, 9, 10, 11, and 14 features, the lowest EER value of 0% and highest SI accuracy of 100% on VCTK data were achieved, while other approaches achieved a lowest EER value of 5% [49] and highest SI accuracy of 98.9% [49] with feature-level fusion, score-level fusion, and i-vector/GMM-UBM on the VCTK voice dataset (Table 26).
5. Voxforge, NIST-2008 and VCTK are medium-size voice databases; therefore, the fusion of 14 features, which is the best common model among the three, can be considered appropriate for medium-size audio datasets.
6. For the voxceleb1 dataset, the least SV EER values of 4.07% and 4.31% and 90% and SI accuracy values of 89.3% are achieved using the fusion of 14 features and 15 features, respectively, with the KNN classifier, while in [55], the best EER of 3.85% is achieved using the x-vector and time delay neural network (TDNN) approach. [56] achieved the best SV EER of 7.8% using a CNN architecture. A total of 1251 speakers were used in [55] and [56], as in the proposed work. In [55], the total number of speaker voice samples is slightly less than that used in the proposed work; hence, the fusion of 14 features can be considered better than the results achieved by [55] and [56] (Table 27)
7. The voxceleb1 dataset has the largest number of speakers among all the datasets used; therefore, the 14 and 15 feature fusion models should be considered suitable for large audio datasets

Table 23 Best feature fusion models on the ELSDSR audio datasets, Approach1 (Proposed best models vs. other best models).

Method	Features used (Model)	Classifier, modeling	Number of speakers	SI accuracy (%)	SV EER (%)
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC (6 features)	KNN	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔMFCC (7 features)	KNN	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔMFCC+ΔRMS+ΔLPC (9 features)	LD	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔMFCC+ΔRMS+ΔMFCC+ΔLPC (10 features)	KNN	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC +entropy Δentropy (12 features)	KNN	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔΔMFCC+ΔLPC+entropy +Δentropy+ΔMFCC+ΔRMS (14 features)	KNN, Ensemble	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy +Δentropy+ΔMFCC+ΔRMS (14 features)	LD, KNN	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy +Δentropy+ΔMFCC+ΔRMS+ΔΔMFCC (15 features)	Ensemble	22 (198 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+centroid+ΔLPC+entropy +Δentropy+ΔMFCC+ΔRMS+ΔΔentropy (15 features)	LD	22 (198 audios)	100	0
Score-level	MFCC, Spectral kurtosis, Spectral skewness,	Random forest (RF)+	22		

fusion [43]	formant extraction, normalized pitch frequency	SVM	(198 audios)	98	2
Feature-level fusion [44]	Deep belief network (DBN) layers, MFCC	GMM-UBM	22 (198 audios)	95	-
Novel pipelined [48]	Gabor filter GF), convolutional neural network (CNN)	SVM, RF, Deep neural network (DNN)	22 (198 audios)	94.8	-

Table 24 Best feature fusion models on the voxforge audio datasets, Approach1 (Proposed best models vs. other best models).

Method	Features used (Model)	Classifier, modeling	Number of speakers	SI accuracy (%)	SV EER (%)
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔMFCC (7 features)	KNN	100 Speakers, (1000 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔ RMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δe ntropy+ΔMFCC+ΔRMS (14 features)	KNN	100 Speakers, (1000 audios)	99.5	0
Feature-level fusion [49]	Two-fold information set (TFIS), MFCC, delta MFCC, delta-delta MFCC	SVM, KNN, Improved Hanman Classifier (IHC)	100 Speakers, (1000 audios)	100	2
Feature-level fusion [51]	MFCC, delta MFCC, delta-delta MFCC	Probabilistic neural network (PNN)	5 Speakers, (750 audios)	94	-
Model fusion [52]	MFCC	GMM, hidden markov model (HMM), generalized fuzzy model (GFM)	100 Speakers, (1000 audios)	93	-

Table 25 Best feature fusion models on the NIST-2008 audio datasets, Approach1 (Proposed best models vs. other best models).

Method	Features used (Model)	Classifier, modeling	Number of speakers	SI accuracy (%)	SV EER (%)
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ Δentropy+ΔLPC+entropy (11 features)	LD	120 (1200 audios)	96.9	0. 2
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔRMS+ΔLPC +entropy+Δentropy (12 features)	LD	120 (1200 audios)	96.9	0. 7
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC +entropy+Δentropy+ΔMFCC+ΔRMS (14 features)	LD	120 (1200 audios)	96.9	0. 3
Score-level fusion [53]	MFCC, power normalized cepstral coefficient (PNCC)	LLR, GMM-UBM	120 (1200 audios)	95.83	
Score-level fusion [54]	PNCC, MFCC	I vector	120 (1200 audios)	96.67	-
Score-level	PNCC, MFCC	GMM-UBM	120	95.83	

fusion [54]			(1200 audios)		-

Table 26 Best feature fusion models on the VCTK audio datasets, Approach1 (Proposed best models vs. other best models).

Method	**Features used (Model)**	**Classifier, modeling**	**Number of speakers**	**SI accuracy (%)**	**SV EER (%)**
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+ΔΔPLP (5 features)	LD	109 Speakers (545 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP (8 features)	KNN	109 Speakers (545 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔLPC (9 features)	LD, KNN	109 Speakers (545 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+entropy+ΔLPC (10 features)	LD	109 Speakers (545 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔR MS+ΔΔPLP+ Δentropy+ΔLPC+entropy (11 features)	LD, Ensemble	109 Speakers (545 audios)	100	0
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC +ΔΔRMS+ΔΔPLP+ΔΔentropy+ΔLPC+entropy+ Δentropy+ΔMFCC+ΔRMS (14 features)	LD, KNN, Ensemble	109 Speakers (545 audios)	100	0
Feature-level fusion [49]	TFIS, MFCC, delta MFCC, delta-delta MFCC	SVM, KNN, IHC	109 Speakers, (545 audios)	98.9	5

Table 27 Best feature fusion models on the voxceleb1 audio datasets, Approach1 (Proposed best models vs. other best models).

Method	Features used (Model)	Classifier, Modeling	Number of speakers	SI accuracy (%)	SV EER (%)
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS +ΔΔPLP+ΔΔentropy+ΔLPC+entropy+Δentropy +ΔMFCC+ΔRMS (14 features)	KNN	1251 Speakers, (153516 audios)	90	4.07
Feature-level fusion [Proposed]	PLP+LPC+ΔPLP+ΔΔLPC+RMS+MFCC+ΔΔRMS +ΔΔPLP+centroid+ΔLPC+entropy+Δentropy +ΔMFCC+ΔRMS+ΔΔentropy (15 features)	KNN	1251 Speakers, (153516 audios)	89.3	4.31
Automated pipelined [56]	Short time magnitude spectrogram	Convolutiona l neural network (CNN)	1251 (153,516 audios)	80.5	-
Score-level fusion [55]	MFCC, deep neural network (DNN)	x vector, attentive static pooling	1246 Speakers, (145058 audios)	-	3.85
Score-level fusion [55]	MFCC, DNN	I vector,	1246 Speakers, (145058 audios)	-	5.39

Automated pipelined [56]	Short time magnitude spectrogram	CNN +Embedding	1251 (153,516 audios)	-	7.8

4.6 Result observation using feature level fusion approach 1

From the results in Tables 23-31, it is observed that feature fusion with delta and delta-delta values generates better SR results than using single features. The fusion of PLP, LPC, ΔPLP, ΔΔLPC, RMS, MFCC, ΔΔRMS, ΔΔPLP, ΔΔentropy, ΔLPC, entropy, Δentropy, ΔMFCC, and ΔRMS (14 features) highlighted in bold red font in the results in Tables 23-27 is the only model that obtains effective results for SI as well as SV on all 5 voice datasets.

Furthermore, when the performance of the three classifiers is compared, it is observed that the KNN classifier performs better on the ELSDSR, voxforge, and voxceleb1 databases (small, medium and large audio datasets), while the LD classifier gives better results on VCTK and NIST-2008 (medium audio datasets). Different classifiers generate different SR results for each voice dataset due to variation in the size of the training/testing datasets. This is why the final effective model with the fusion of 14 features is generated by different classifiers for each voice dataset. Additionally, the results generated by the proposed models are better than the other results; hence, the selection of the 2 best models at each step can be considered an effective way to produce the best SR results.

Fig.23 to fig. 26 shows SI accuracy using various feature fusion model for clean voice data ELSDSR, VCTK, NIST-2008 and voxforge data respectively.

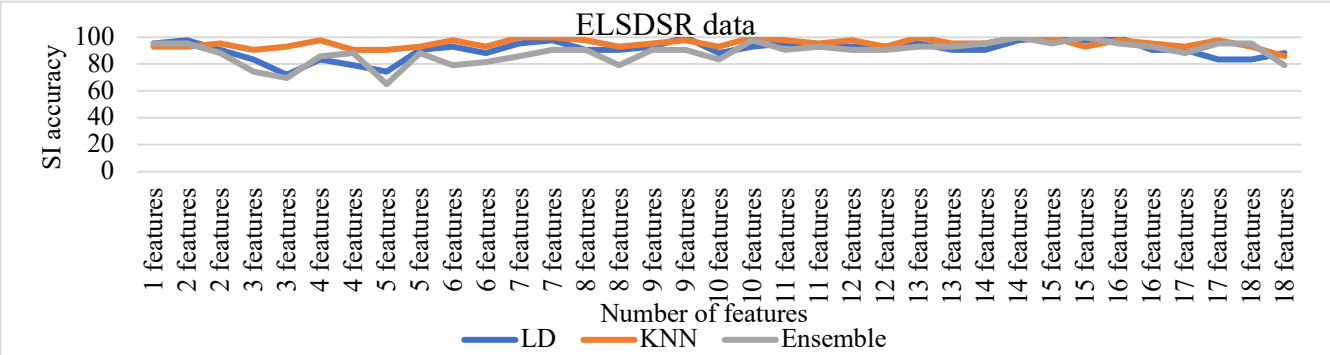


Fig.23 Change in SI accuracy using various feature fusion and classifier for ELSDSR data.

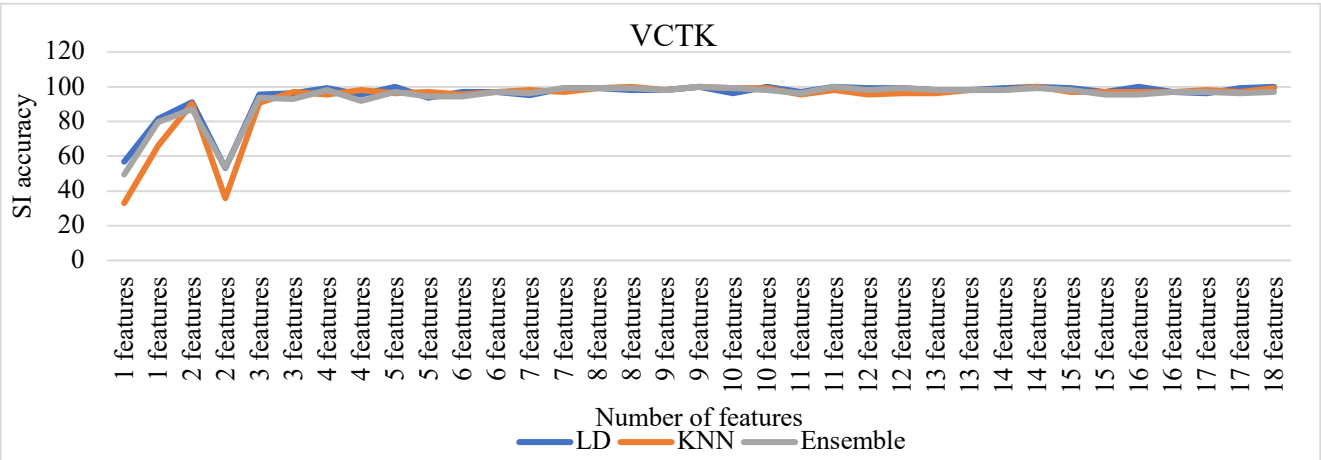


Fig.24 Change in SI accuracy using various feature fusion and classifier for VCTK data.

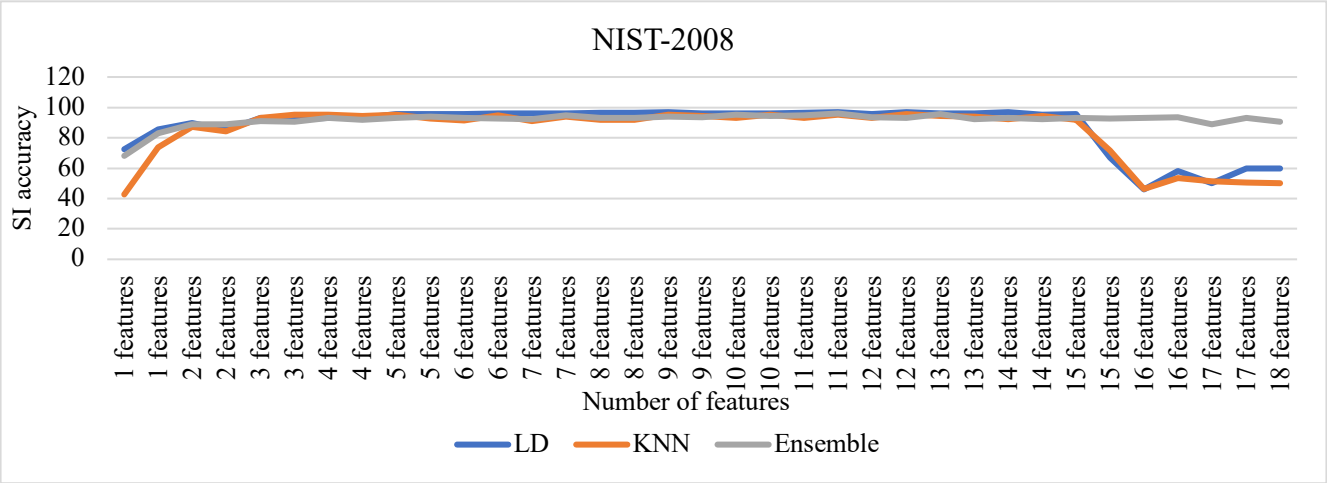


Fig.25 Change in SI accuracy using various feature fusion and classifier for NIST-2008 data.

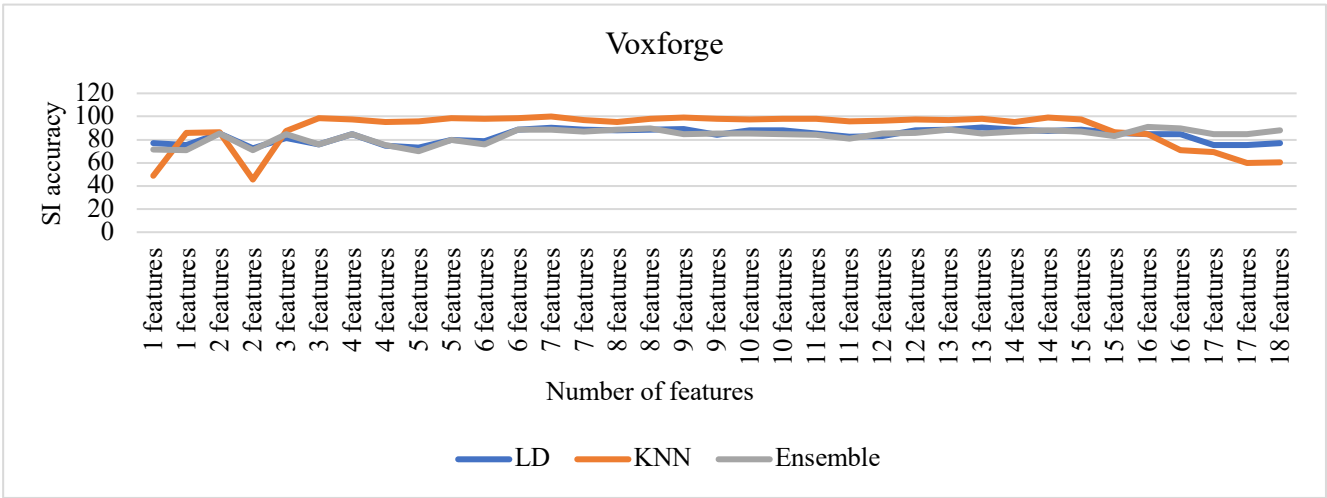


Fig.26 Change in SI accuracy using various feature fusion and classifier for voxforge data.

4.7 Result discussion for feature level fusion (approach 1), dimension reduction (approach 2) and feature optimization (approach 3) for noisy data

Under this section we discuss result obtained using feature level fusion (approach 1), dimension reduction technique (approach 2) and feature optimization (approach 3) for noisy datasets. Table 28 displays the best model results using feature level fusion approach 1 for all databases. Tables 29, 30, and 31 show SI accuracy and EER values for 18-feature combinations in babble noise, white noise, and voxceleb1 data, respectively. Table 32 presents the best results achieved with dimension reduction techniques (approach 2) for the 18-feature combination model, while table 33 shows the best results using the feature optimization method. The proposed work achieves the highest SI accuracy and least SV EER across all three approaches, demonstrating the effectiveness of the model in various scenarios and datasets.

4.7.1 Best results using feature level fusion (approach 1)

Table 28 presents the best results achieved using feature level fusion approach 1 for all databases. It shows the highest SI accuracy and EER values for each database and the corresponding fusion of features. For the TIMIT babble noise database, the best SI accuracy and EER values achieved using the LD classifier are 92.7% (120 speakers) and 89.3% (630 speakers), with EER values of 4.4% and 2.2%, respectively. The models with fusion of 12 features and 14 features give the best results for 120 and 630 speakers, respectively.

Similarly, for the TIMIT white noise database, the best SI accuracy and EER values using the LD classifier are 93.3% (120 speakers) and 79.4% (630 speakers), with EER values of 1.1% and 2.4%, respectively. The models with fusion of 11 features and 12 features give the best results for 120 and 630 speakers, respectively.

For the voxceleb1 database, the highest SI accuracy achieved is 90%, and the least EER is 4.07% [1] is achieved using fusion of 14 features with KNN classifier.

Tables 29, 30, and 31 show the SI accuracy and SV EER using all 18 feature combinations for LD, KNN, and ensemble classifiers with babble noise, white noise, and voxceleb1 as input databases, respectively. Figures 27 to 31 demonstrate the improvement in SI accuracy when using more feature fusion compared to using single features for all the databases used mainly foy noisy speech datasets.

4.7.2 Computation time comparison with feature-level fusion (approach 1)

Comparing the results from different databases and feature sets, we observed the following trends:

1.For the 120-speaker babble noise database:

- Using all 18 features with the LD classifier, the training time was 5.8 seconds, and the testing time was 0.8 seconds (Table 29).
- Utilizing 12 features with the LD classifier resulted in better performance, with reduced training time of 4.9 seconds and testing time of 0.8 seconds (Table 28).

2. For the 630-speaker babble noise database:

- Using all 18 features with the LD classifier, the training time was 10.9 seconds, and the testing time was 4.7 seconds (Table 29).
- Utilizing 14 features with the LD classifier achieved better performance, with reduced training time of 8.9 seconds and testing time of 0.9 seconds (Table 28).

3. In the case of the 120-speaker white noise database:

- Using all 18 features with the LD classifier, the training time was 8.5 seconds, and the testing time was 1.7 seconds (Table 30).
- Utilizing 11 features with the LD classifier led to improved performance, with reduced training time of 4.8 seconds and testing time of 1.2 seconds (Table 28).

4. For the 630-speaker white noise database:

- Using all 18 features with the LD classifier, the training time was 22.5 seconds, and the testing time was 4.7 seconds (Table 30).
- Utilizing 14 features with the LD classifier resulted in better performance, with reduced training time of 14.9 seconds and testing time of 3.2 seconds (Table 28).

5. When dealing with the voxceleb1 database:

- Using all 18 features with the KNN classifier, the training time was 2090 seconds, and the testing time was 50.9 seconds (Table 31).
- Utilizing 14 features with the KNN classifier achieved better performance, with reduced training time of 1458.6 seconds and testing time of 48.79 seconds (Table 28).

Note: Only the best classifier's training and testing time are included and compared in table 28.

Table 28 Best result using feature fusion method (Approach 1).

Features used (Model)	Classifier, Modelling	Database	Number of speakers	Number of Feature vectors	Training Time (sec)	Testing Time (sec)	SI accuracy (%)	SV EER (%)
LPC+PLP+ΔMFCC +ΔPLP+MFCC +ΔΔentropy+Δentropy +ΔΔRMS+entropy		TIMIT babble						

+RMS+ΔLPC+ΔΔPLP	LD	noise	120	96	4.9	0.8	92.7	4.4
LPC+PLP+ΔMFCC +ΔPLP+MFCC +ΔΔentropy+Δentropy +ΔΔRMS+entropy +RMS+ΔLPC +ΔΔPLP+ΔRMS +ΔΔLPC	LD	TIMIT babble noise	630	110	8.9	0.9	89.3	2.2
LPC+PLP+ΔMFCC +ΔPLP+MFCC +ΔΔentropy+Δentropy +ΔΔRMS+entropy +RMS+ΔΔPLP	LD	TIMIT white Noise	120	83	4.8	1.2	93.3	1.1
LPC+PLP+ΔMFCC +ΔPLP+MFCC +ΔΔentropy+Δentropy +ΔΔRMS+entropy +RMS+ΔLPC+ΔΔPLP	LD	TIMIT white Noise	630	96	14.9	3.2	79.4	2.4
PLP + LPC + ΔPLP + ΔΔLPC + RMS + MFCC + ΔΔRMS + ΔΔPLP + ΔΔentropy + ΔLPC + entropy + Δentropy + ΔMFCC + ΔRMS	KNN	Voxceleb1	1251	110	1458.6	48.79	90	4.07

Table 29 Best SI accuracy and EER using all feature model for all database for babble noise (126 features vectors).

Classifier, Modelling	Total number of speakers	Training Time (seconds)	Testing Time (seconds)	SI accuracy (%)	SV EER (%)
LD	120	5.8	0.8	89.8	1.09
KNN	120	2.24	0.9	79.8	0.77
Ensemble	120	5.7	1.6	85.8	30
LD	630	10.9	4.7	89.9	1.1
KNN	630	2.8	2.9	82.9	0.14
Ensemble	630	134	11.3	81.3	1.02

Table 30 Best SI accuracy and EER using all feature model for all database for white noise (126 features vectors).

Classifier, Modelling	Total number of speakers	Training Time (seconds)	Testing Time (seconds)	SI accuracy (%)	SV EER (%)
LD	120	8.5	1.7	86.9	0.9
KNN	120	9.9	1.6	79.4	1.2
Ensemble	120	9.9	2.3	84.8	1.5
LD	630	22.5	4.7	79.2	3
KNN	630	4.3	3.2	73.4	0.16
Ensemble	630	181.7	10.9	73.1	4.2

Table 31 Best SI accuracy and EER using all feature model for all database for voxceleb1 noise (126 features vectors).

Classifier, Modelling	Training Time (seconds)	Testing Time (seconds)	SI accuracy (%)	SV EER (%)
LD	2206	28.9	70.9	15.3
KNN	2090.9	50.9	89.7	4.5
Ensemble	11108	256.8	63.7	31.2

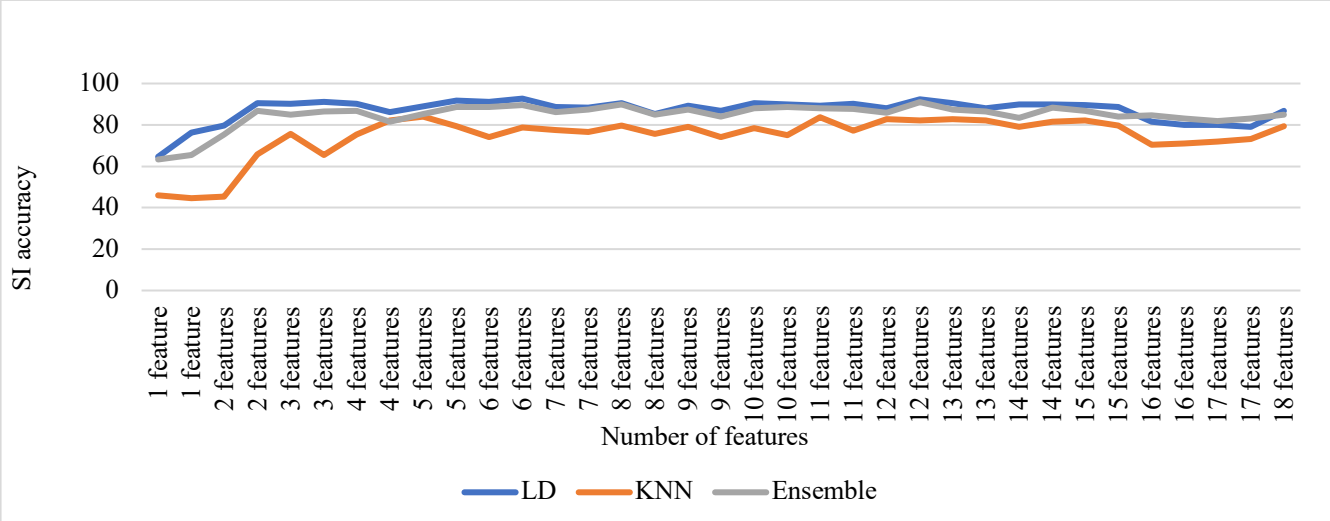


Fig.27 Change in SI accuracy using various feature fusion and classifier for babble noise data (120 speakers) (Approach 1).

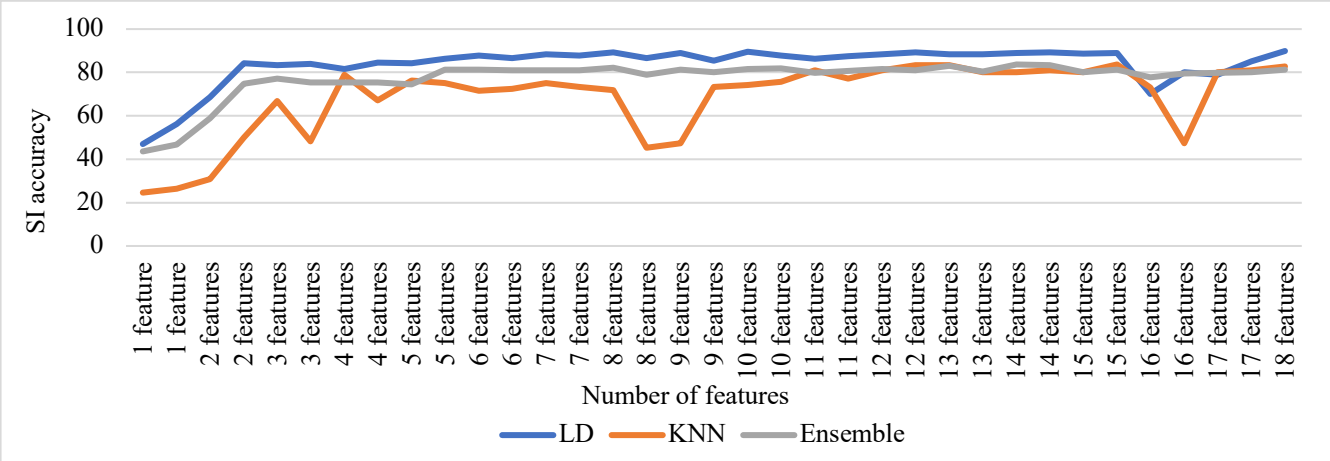


Fig.28 Change in SI accuracy using various feature fusion and classifier for babble noise data (630 speakers) (Approach 1).

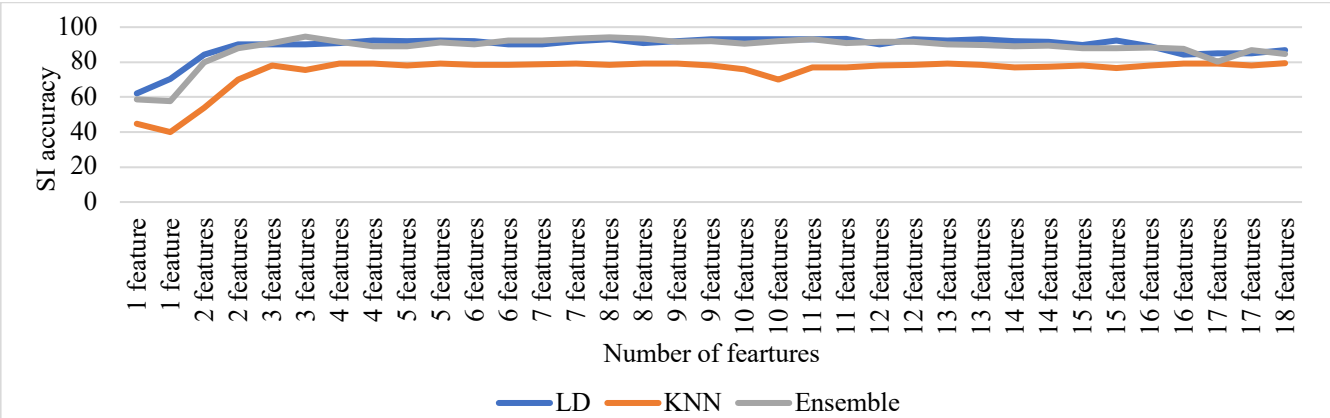


Fig.29 Change in SI accuracy using various number of features combination for white noise 120 speakers (Approach 1).

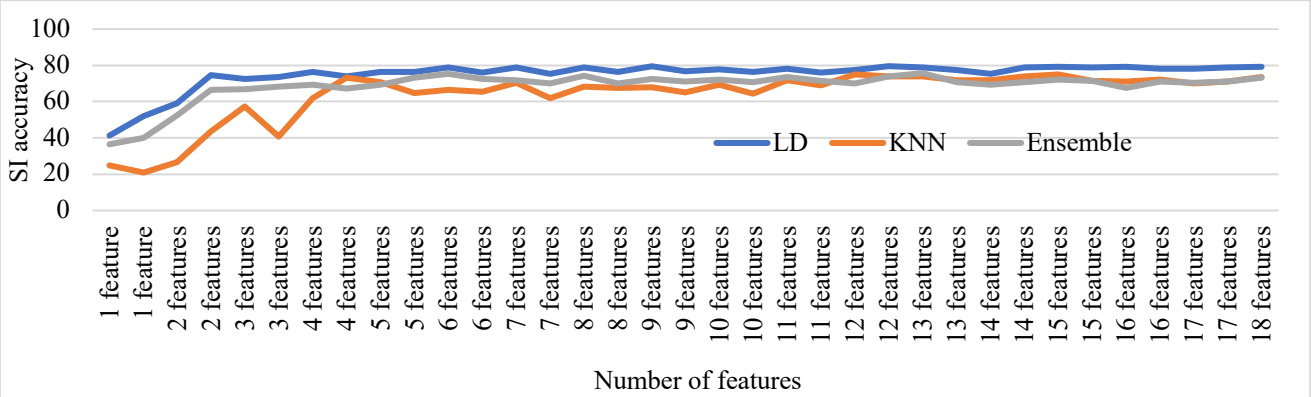


Fig.30 Change in SI accuracy using various number of features combination for white noise 630 speakers (Approach 1).

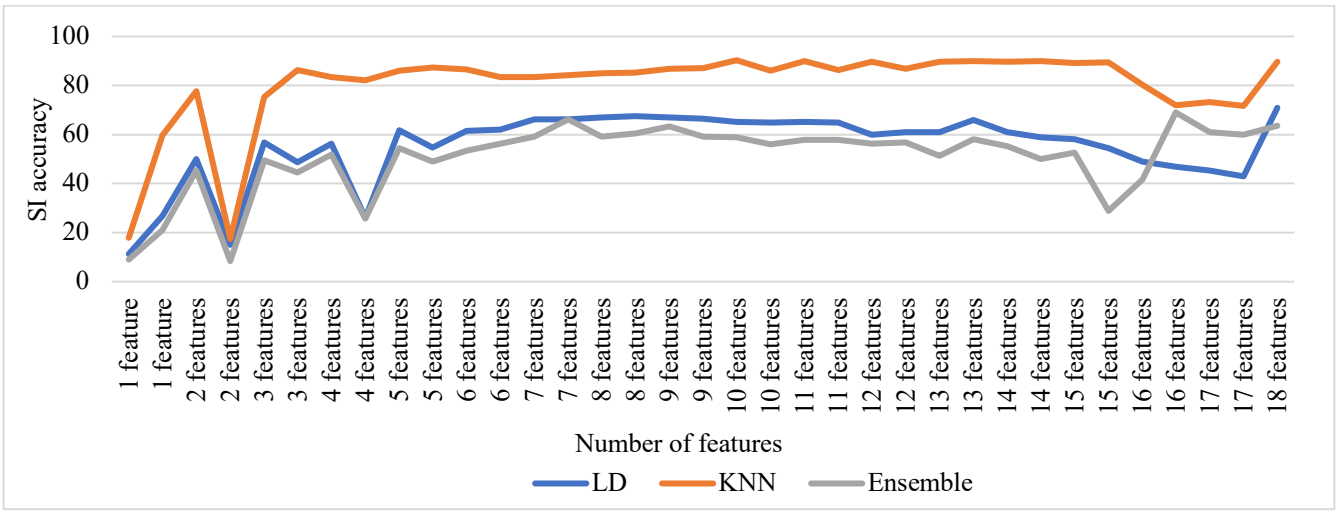


Fig.31 Change in SI accuracy using various number of features combination for voxceleb1 (Approach 1).

4.7.3 Best results using dimension reduction technique (approach 2)

Table 32 displays the best results achieved using principal component analysis (PCA) for dimension reduction and includes the computation timing for training and testing. Since PCA outperforms ICA (independent component analysis), only PCA results are shown. Figure 32, figure 33, and figure 34 illustrate the variation in SI accuracy using different PCA/ICA feature dimensions and the SI accuracy using all 18 feature combinations for babble noise, white noise, and voxceleb1 databases.

For the TIMIT babble noise database with 120 speakers, the best SI accuracy of 89.9% and EER of 0.9% is achieved using 126 PCA feature vectors with the LD classification method. For TIMIT babble noise with 630 speakers, the best SI accuracy of 90.6% and EER of 0.69% is achieved using 80 PCA feature vectors with the KNN classifier.

In the case of TIMIT white noise with 120 speakers, the best SI accuracy of 93.3% and EER of 0.58% is achieved using 100 PCA feature vectors with the KNN classification. For TIMIT white noise with 630 speakers, the best SI accuracy of 81.4% and EER of 0.13% is achieved using 126 PCA feature vectors with the KNN classification.

For the voxceleb1 database, the best SI accuracy of 94.7% and the least EER of 2.2% are achieved using 126 feature vectors with the KNN classifier. For voxceleb1 data we can observe that performance improves using dimension reduction techniques than approach 1 [1] .

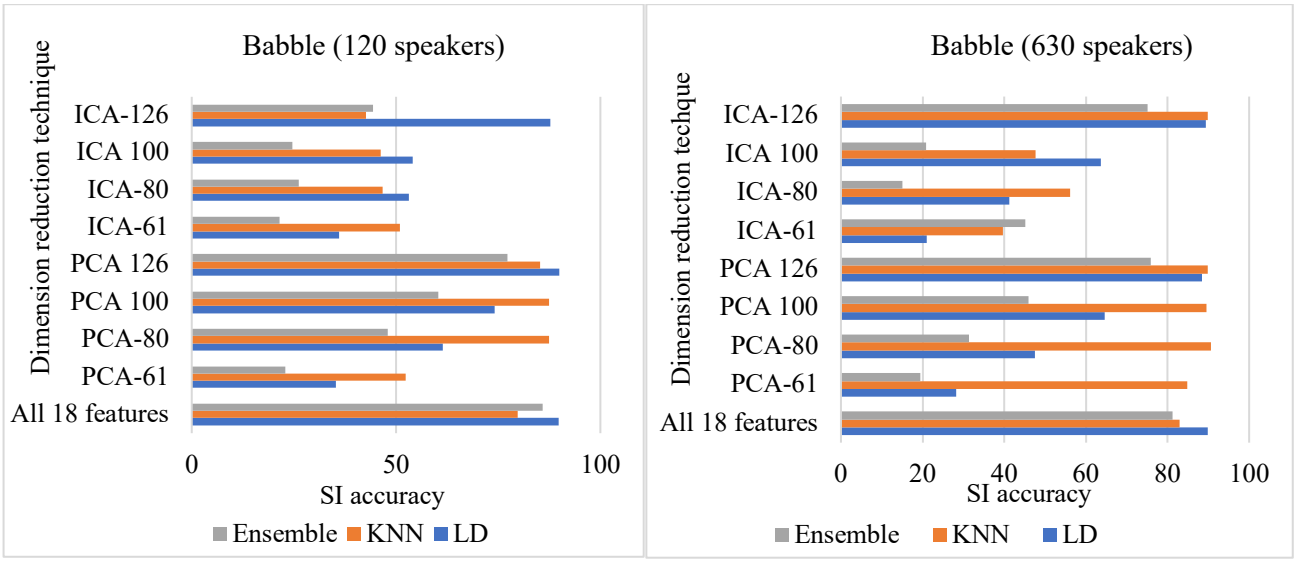


Fig.32 SI accuracy for different classification methods using dimension reduction techniques with TIMIT babble noise dataset (Approach 2).

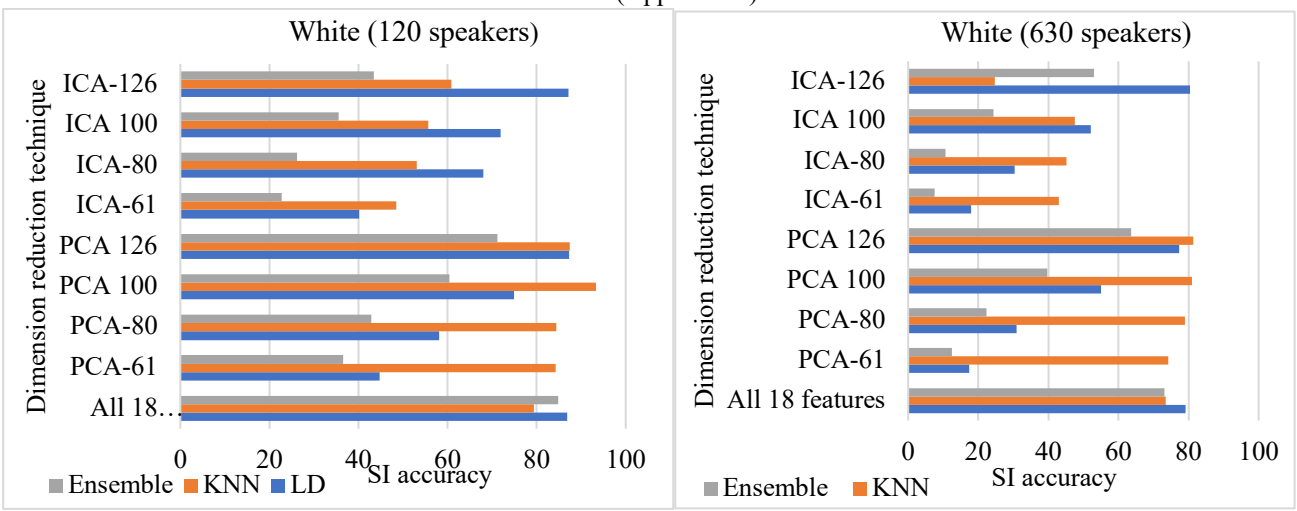


Fig.33 SI accuracy for different classification methods using dimension reduction techniques with TIMIT white noise dataset (Approach 2).

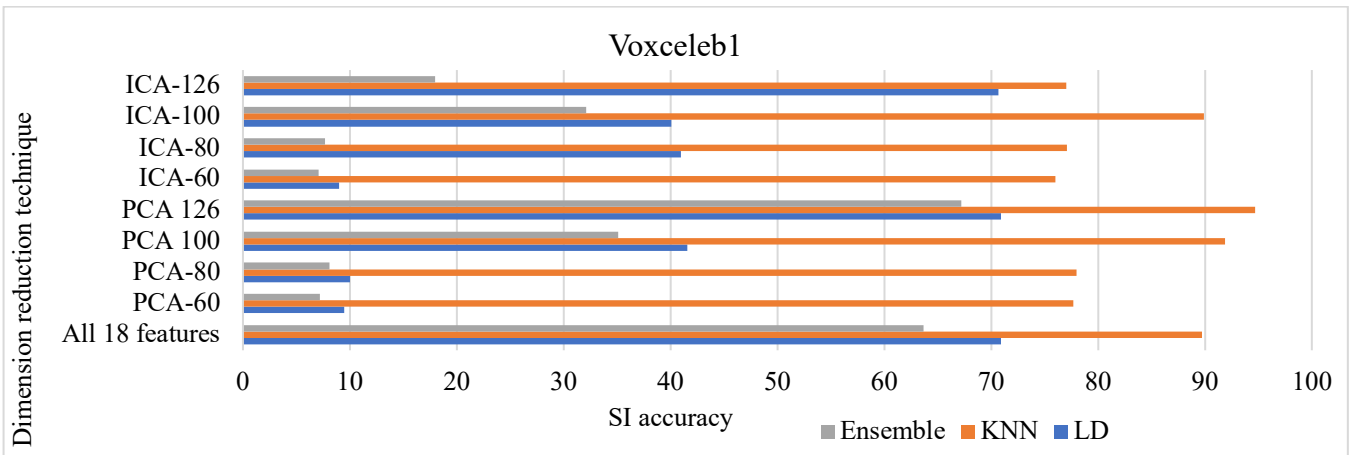


Fig.34 SI accuracy for different classification methods using dimension reduction techniques with voxceleb1 dataset (Approach 2).

4.7.4 Computation time with dimension reduction (approach 2)

Upon analyzing the results from Table 32 and the preceding tables, notable enhancements were observed in speaker recognition using PCA dimension reduction:

1. For the 120-speaker babble noise database:

- Utilizing all 18 features with the LD classifier resulted in a training time of 5.8 seconds and a testing time of 0.8 seconds (Table 29).
- By employing PCA dimension reduction approach 2 with 126 PCA feature vectors, we achieved better performance, with reduced training time of 2.5 seconds and testing time of 0.7 seconds using the LD classifier.

2. For the 630-speaker babble noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 2.8 seconds and a testing time of 2.9 seconds (Table 29).
- Applying PCA dimension reduction technique with 80 PCA feature vectors led to slightly better performance, with reduced training time of 2.7 seconds and testing time of 0.9 seconds (Table 32) using the KNN classifier.

3. In the case of the 120-speaker white noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 9.9 seconds and a testing time of 1.6 seconds (Table 30).
- By using PCA dimension reduction technique with 100 PCA feature vectors, we achieved better performance, with reduced training time of 5.8 seconds and testing time of 1.2 seconds (Table 32) using the KNN classifier.

4. For the 630 speaker white noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 4.3 seconds and a testing time of 3.2 seconds (Table 30).
- Implementing PCA dimension reduction technique with 126 PCA feature vectors resulted in better performance, with reduced training time of 3.08 seconds and testing time of 2.9 seconds (Table 32) using the KNN classifier.

5. When dealing with the voxceleb database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 2090 seconds and a testing time of 50.9 seconds (Table 31).
- Adopting dimension reduction technique, specifically PCA, led to better performance, with reduced training time of 1646 seconds and testing time of 72.6 seconds (Table 32) using the KNN classifier.
- Overall, PCA proved to be the superior dimension reduction technique, demonstrating improved speaker recognition accuracy and significantly faster computation times in various noise conditions and different speaker databases. Only the best classifier's training and testing times are included and compared in Table 32.

Table.32 Best model using dimension reduction (Approach 2).

Method	Classifier	Feature used	Database	Number of speakers	Number of Feature vectors	Training Time (sec)	Testing Time (sec)	SI accuracy (%)	SV EER (%)
PCA	LD	All 18	TIMIT babble noise	120	126	2.5	0.7	89.9	0.9
PCA	KNN	ALL 18	TIMIT babble noise	630	80	2.7	0.9	90.6	0.69
PCA	KNN	All 18	TIMIT white noise	120	100	5.8	1.2	93.3	0.58
PCA	KNN	All 18	TIMIT white noise	630	126	3.08	2.9	81.4	0.13
PCA	KNN	ALL18	Voxceleb1	1251	126	1646	72.6	94.7	2.2

4.7.5 Optimal results achieved with feature optimization technique (approach 3)

Table 33 presents the most promising outcomes attained through feature optimization using proposed methods. For babble noise data with 120 and 630 speakers, the best accuracy achieved was 85.6% and 93.5%, with EER values of 0.7% and 0.13%, respectively, using the KNN classification. The feature vectors were reduced to 81 for the TIMIT babble noise 120 speakers and 90 for the TIMIT babble noise 630 speakers using the PCA-GA approach 3.

Regarding TIMIT white noise data, the best accuracy achieved was 87.9% and 83.5% for 120 and 630 speakers, respectively, with the best EER of 0.8% and 0.13%, respectively. The optimal performance was achieved using the PCA-MPA feature optimization method with the KNN classifier, with reduced feature vectors of 103 and 112 for 120 and 630 speakers for TIMIT white noise data, respectively.

For voxceleb1 data, the PCA-MPA feature optimization method with 112 reduced feature vectors and the KNN classifier achieved the best accuracy of 95.2% and EER of 1.8%. In figures 35, 36, and 37, a comparison between different feature optimization approaches, including PCA-GA, PCA-MPA, ICA-GA, ICA-MPA, feature-GA, and feature-MPA, illustrates that PCA-GA and PCA-MPA outperformed ICA-GA, ICA-MPA, feature-GA, and feature-MPA models for babble noise, white noise and voxceleb1 data respectively.

The results indicate that PCA-based optimization approaches, namely PCA-GA and PCA-MPA, demonstrate superior performance in comparison to other methods.

Table.33 Best model using feature optimization (Approach 3).

Method	Classifier	Feature used	Database	Number of speakers	Number of Feature vectors	Training Time (sec)	Testing Time (sec)	SI accuracy (%)	SV EER (%)
PCA-GA	KNN	All 18	TIMIT babble noise	120	81	1.9	0.9	85.6	0.7
PCA-GA	KNN	ALL 18	TIMIT babble noise	630	90	2.4	0.8	93.5	0.13
PCA-MPA	KNN	All 18	TIMIT white noise	120	103	2.7	1.2	87.9	0.8
PCA-MPA	KNN	All 18	TIMIT white noise	630	112	1.7	1.8	83.5	0.13
PCA-MPA	KNN	All 18	Voxceleb1	1251	112	1374.3	42.5	95.2	1.8

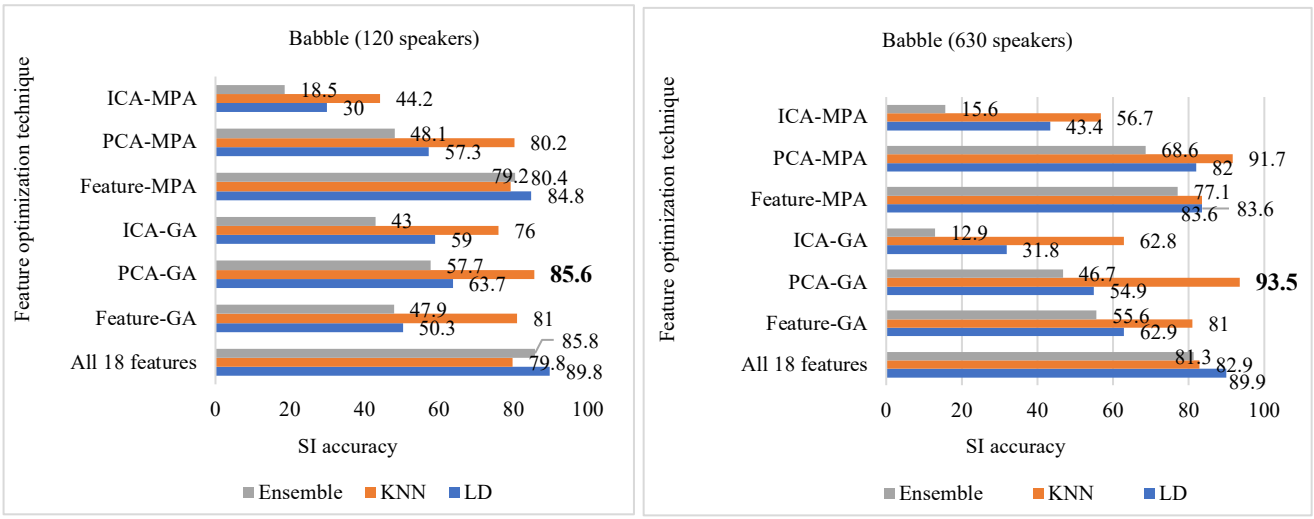


Fig.35 SI accuracy for different classification methods and datasets using feature selection techniques with TIMIT babble noise (Approach 3).

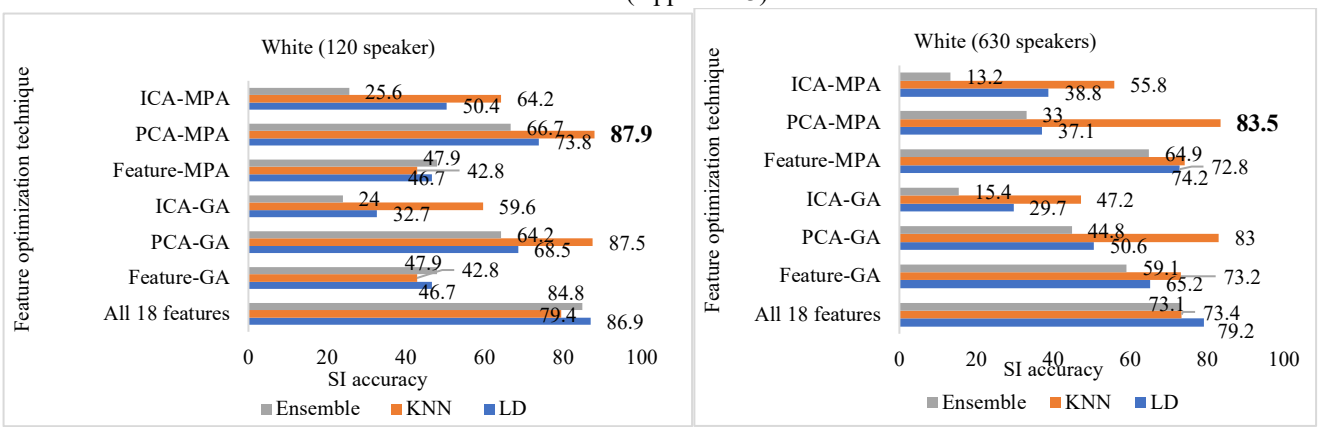


Fig.36 SI accuracy for different classification methods and datasets using feature selection techniques with TIMIT white noise (Approach 3).

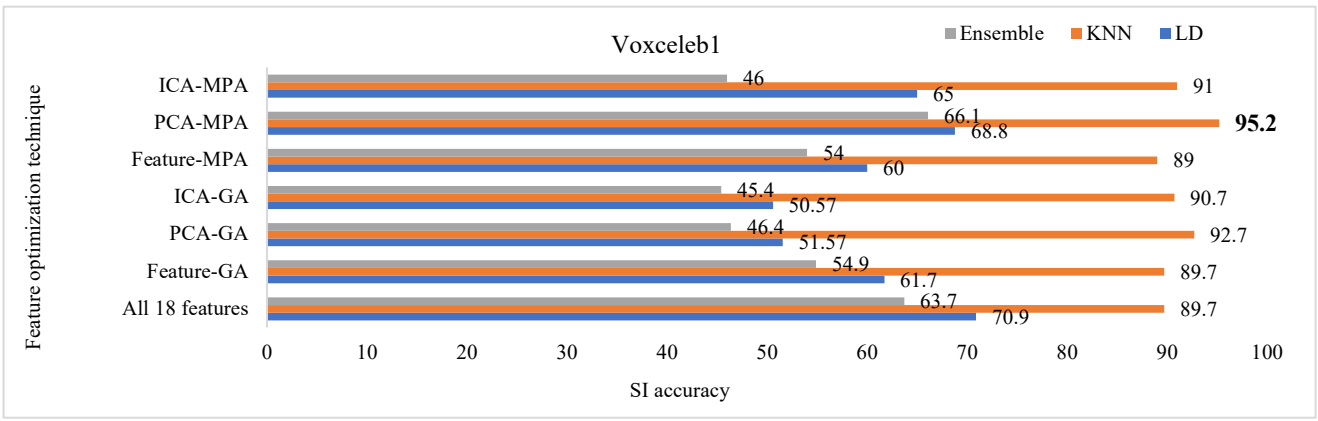


Fig.37 SI accuracy for different classification methods and datasets using feature selection techniques with voxceleb1 (Approach 3).

4.7.6 Computation time with feature optimization methods (approach 3)

The comparison between Table 33 and the previous tables reveals the following enhancements achieved through feature optimization:

1. For the 120-speaker babble noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 2.24 seconds and a testing time of 0.9 seconds (Table 29).
- By employing PCA-GA feature optimization approach 3 with 81 PCA-GA feature vectors, we achieved better performance, with reduced training time of 1.9 seconds and testing time of 0.9 seconds using the KNN classifier (Table 33).

2. For the 630-speaker babble noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 2.8 seconds and a testing time of 2.9 seconds (Table 29).
- Applying PCA-GA feature optimization approach 3 with 90 PCA-GA feature vectors led to slightly better performance, with reduced training time of 2.4 seconds and testing time of 0.8 seconds using the KNN classifier (Table 33).

3. In the case of the 120-speaker white noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 9.9 seconds and a testing time of 1.6 seconds (Table 30).
- By using PCA-MPA dimension reduction technique, we achieved better performance, with reduced training time of 2.7 seconds and testing time of 1.2 seconds using the KNN classifier with 103 PCA-MPA feature vectors (Table 33).

4. For the 630-speaker white noise database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 4.3 seconds and a testing time of 3.2 seconds (Table 30).
- Implementing PCA-MPA feature optimization approach, we achieved better performance, with reduced training time of 1.7 seconds and testing time of 1.8 seconds using the KNN classifier with 112 PCA-MPA feature vectors (Table 33).

5. When dealing with the voxceleb1 database:

- Utilizing all 18 features with the KNN classifier resulted in a training time of 2090 seconds and a testing time of 50.9 seconds (Table 31).
- Adopting dimension reduction techniques, we achieved better performance, with reduced training time of 1374.3 seconds and testing time of 42.5 seconds using the KNN classifier (Table 33).

Overall, the results demonstrate that feature optimization techniques, specifically PCA-GA and PCA-MPA, significantly enhance SI accuracy and reduce computation time for noisy and multimedia voice data. Only the best classifier's training and testing times are included and compared in Table 33.

4.8 System configuration

g-gear neo gx9j-c181/zt GPU is used for the computation of training and testing time on matlab software.

4.9 Comparing proposed work with existing approach.

Table 34 to Table 36 present the best results achieved by the three proposed approaches for all datasets used, and we compare these results with other best results obtained using the same input data.

4.9.1. TIMIT babble noise (120 speakers)

The highest SI accuracy of 92.7% is achieved using feature level fusion (Approach 1) with 12 features and LD classification .The least EER of 0.13% is achieved using PCA-GA feature selection (Approach 3) .In comparison, [61] and [62] achieved the best EER of 4.3% and 6.39% using GMM and i-vector approaches with 368 and 630 speakers, respectively, for babble noise data (Table 34).

4.9.2 TIMIT babble noise (630 speakers)

The best SI accuracy and EER of 93.5% and 0.13% are achieved using PCA-GA feature selection with the KNN classifier (Table 34). In comparison, [61] and [62] achieved EERs of 4.3% and 6.39% using GMM and i-vector approaches. While [87] achieves 77.51% SI accuracy using deep learning method for TIMIT clean dataset with 630 speakers (Table 34).

Table 34 Result comparison table for TIMIT Babble noise.

Method	Features used (Model)	Classifier, Modelling	Speech Database	Number of speakers	Number of Feature vectors	Feature selection technique	SI accuracy (%)	SV EER (%)
Feature level fusion (approach 1) (Proposed)	LPC+PLP+ΔMFCC+ΔPLP +MFCC +ΔΔentropy+ Δentropy+ ΔΔRMS+ entropy+RMS+ ΔLPC+ΔΔPLP	LD	TIMIT-babble noise, 30DB	120	96	Non	92.7	1.3
Feature selection (approach 2) (proposed)	All 18	KNN	TIMIT-babble noise, 30DB	120	81	PCA-GA	85.6	0.7
Feature selection (approach 2) (proposed)	All 18	KNN	TIMIT-babble noise, 30DB	630	90	PCA-GA	93.5	0.13
Spectral subtraction [61]	IMFCC	GMM	TIMIT babble noise-10DB	368	36	Non	-	4.3
New Feature extraction [62]	MGCC	I-vector	TIMIT babble noise 20 DB	630	13	LDA	-	6.39
Deep learning [87]	MFCC	CNN	TIMIT-Clean	630	13	-	77.51	

4.9.3 TIMIT white noise (120 speakers)

The highest SI accuracy of 93.3% and least EER of 0.58% are achieved using PCA dimension reduction (Approach 2) with 100 feature vectors (Table 35). In comparison, [53] and [54] achieved the highest accuracy of 75.83% and 79.17%, respectively, using the score level fusion method with the same number of 120 speakers and 30 dB noisy data (Table 35).

4.9.4 TIMIT white noise (630 speakers)

The highest SI accuracy of 83.5% and least EER of 0.13% are achieved using PCA-MPA feature optimization (Approach 3) with 112 selected feature vectors and the KNN classifier (Table 33). In comparison, [60], [61], and [62] achieved accuracy of 63%, EER of 7.1%, and 8%, respectively, using GMM and i-vector approaches. While [87] achieves 77.51% SI accuracy using deep learning method for TIMIT clean dataset with 630 speakers (Table 35).

Table.35 Result comparison table for TIMIT white noise.

Method	Features used (Model)	Classifier, Modelling	Speech Database	Number of speakers	Number of Feature vectors	Dimension reduction technique	SI accuracy (%)	SV EER (%)
Dimension reduction (approach 2) (proposed)	All 18 features	KNN	TIMIT-white noise, 30DB	120	100	PCA	93.3	0.58
Feature selection method (approach 3) (proposed)	All 18 features	KNN	TIMIT-white noise, 30DB	630	112	PCA-MPA	83.5	0.13
Score level fusion [53]	MFCC, PNCC	gmm-ubm, llr classifier	TIMIT awgn and G.712 noise 30 DB	120	16	Non	75.83	-
Score level fusion [54]	MFCC, PNCC	gmm-ubm maximum likelihood ratio	TIMIT AWGN-30DB	120	16	Non	79.17	-
ICA feature extraction [60]	ICA	GMM	TIMIT-white noise, 20DB	100	36	ICA	63	-
Spectral subtraction [61]	IMFCC	GMM	TIMIT white noise-10 DB	368	36	Non		7.1
New Feature extraction [62]	MGCC	I-vector	TIMIT white noise-(20 DB)	630	13	LDA		8
Deep learning [87]	MFCC	CNN	TIMIT-Clean	630	13	-	77.51	

4.9.5 Voxceleb1 data (largest dataset)

The best SI accuracy of 95.2% and EER of 1.8% are achieved using PCA-MPA feature optimization (Approach 3) with 112 feature vectors and the KNN classifier. In comparison, [55], [56], [88], [89] and [90] achieved EERs of 3.85%, 7.8%, 3.1%, 4.46% and 2.52% respectively, using x-vectors, i-vector methods, CNN, x-vectors, temporal average pooling techniques and x vector+r vector approaches (Table 36).

Table.36 Result comparison table for voxceleb1.

Method	Features used (Model)	Classifier, Modelling	Number of Feature vectors	Number of speakers	Dimension reduction technique	SI accuracy (%)	SV EER (%)
Feature selection (approach 3)	All 18	KNN		1251 Speakers, (153516		95.2	1.8

(Proposed)			112	audios)	PCA-MPA		
Score level fusion [55]	MFCC, Deep Neural Network (DNN)	x vector, attentive static pooling	60	1246 Speakers, (145058 audios)	-	-	3.85
Score level fusion [55]	MFCC, DNN	I vector,	60	1246 Speakers, (145058 audios)	-	-	5.3
Automated pipelined [56]	Short time magnitude spectrogram	CNN +Embedding	13	1251 (153,516 audios)	-	-	7.8
DNN [88]	DNN	x-vector	-	1251	-	-	3.1
temporal average pooling [89]	MFCC	A-SOFTMAX	60	1251	-	-	4.46
Temporal average pooling [89]	MFCC	Convolutional neural network with local discriminant embedding CNN-LDE	60	1251	-	89.9	-
FTDNN +Res2Net- [90]	DNN	x-vector,r-vector	80	1251			2.52

4.10 Overall comparison

4.10.1 Summary of Feature Fusion Approach 1 on Clean Voice Datasets: Voxceleb1, ELSDSR, Voxforge, VCTK, and NIST-2008

We initially worked on five types of clean voice datasets: Voxceleb1, ELSDSR, Voxforge, VCTK, and NIST-2008. Our goal was to find the best common feature combination for all the datasets used. We found that the combination of the following 14 features: PLP, LPC, ΔPLP, ΔΔLPC, RMS, MFCC, ΔΔRMS, ΔΔPLP, ΔΔentropy, ΔLPC, entropy, Δentropy, ΔMFCC, and ΔRMS is suitable for all the datasets. This is evident from figures 38 and 39.

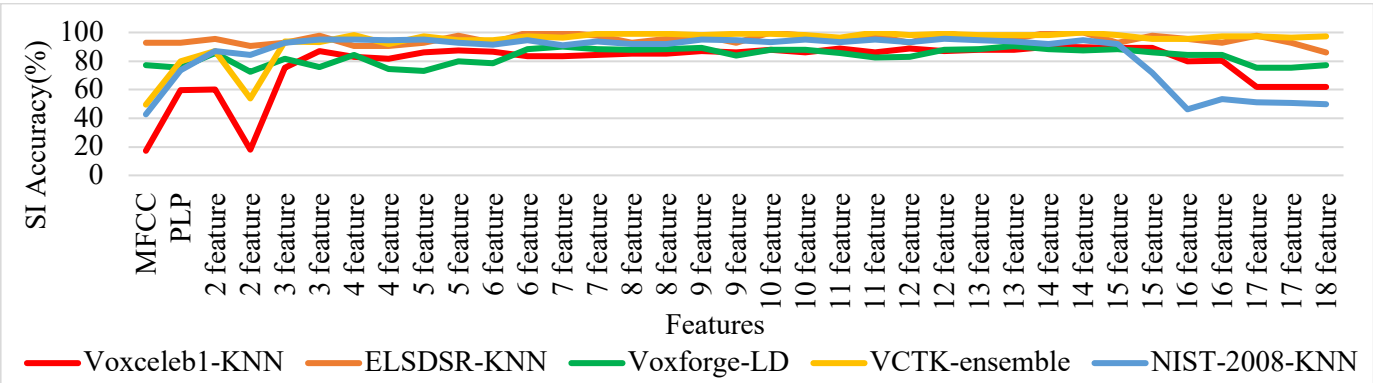


Fig.38 Speaker identification (SI) performance on the ELSDSR, Voxforge, VCTK, NIST-2008 and voxceleb1 audio datasets.

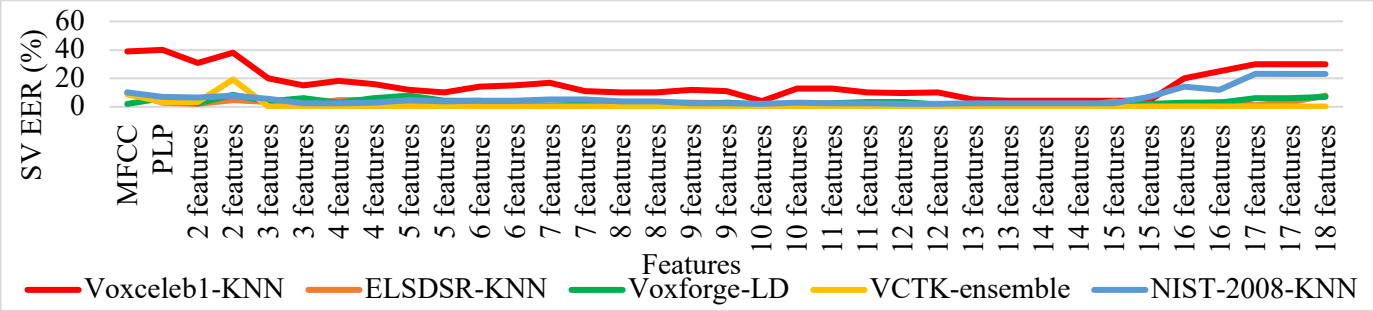


Fig.39 Speaker Verification (SI) performance on the ELSDSR, Voxforge, VCTK, NIST-2008 and voxceleb1 audio datasets.

4.10.2 Summary of All Three Approaches for Noisy Datasets: Feature Fusion (Approach 1), Dimension Reduction (Approach 2), and Feature Optimization (Approach 3) on TIMIT White Noise, TIMIT Babble Noise, and Voxceleb1 Data

The proposed approaches consistently outperform other methods, with feature optimization (Approach 3) yielding the best results most of the time. Feature optimization techniques not only reduce computation time but also enhance KNN classification when combined with dimension reduction (Approach 2) and feature optimization (Approach 3). Meanwhile, linear discriminant (LD) and ensemble classification perform better with Feature Level Fusion (Approach 1). The application of GA and MPA feature optimization on PCA features significantly contributes to the field of speaker recognition for both noisy and multimedia databases. Figure 40 illustrates the result comparison of Babble Noise, White Noise, and Voxceleb data using all approaches with the KNN classifier. It is evident that overall, Approach 3 (feature optimization) performs the best across all noisy data, while Approach 2 (dimension reduction) and Approach 1 (feature fusion) follow as the second and third best, respectively. In conclusion, the proposed approaches demonstrate remarkable performance improvements compared to existing methods across different datasets, showcasing the effectiveness of feature optimization techniques in speaker recognition tasks.

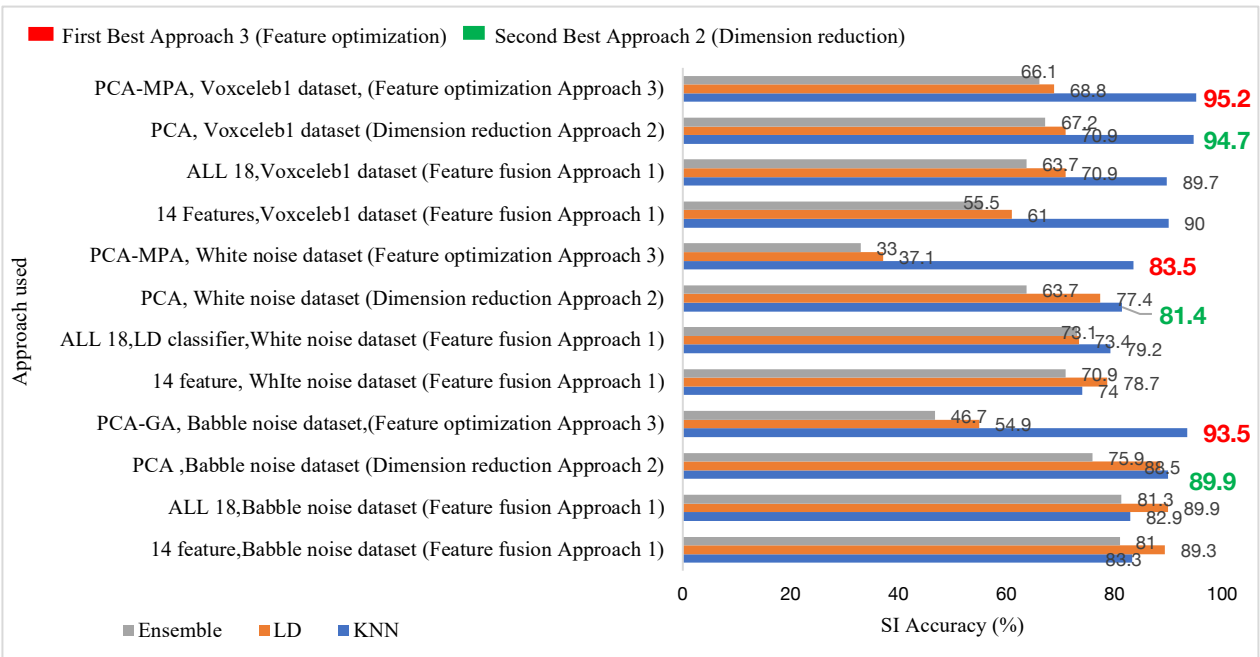


Fig.40 Result Comparison of Babble Noise, White Noise, and Voxceleb1 Data using all approaches.

4.11 Comparison of training and testing computation time using all 3 proposed approaches.

Observing Figure 38, Figure 39, and Figure 40, we can determine the best model training and testing times for babble noise, white noise, and voxceleb1 data using various classification methods and our three proposed approaches. Upon comparing the computation times of all three proposed approaches with the computation time of the model utilizing all 18 features, an important finding emerges: Approach 3, which involves feature optimization, consistently exhibits faster computation times compared to the other methods. In light of this, it becomes evident that the feature optimization method with PCA-GA and PCA-MPA (Approach 3) holds significant utility for enhancing the speed and efficiency of speaker recognition systems.

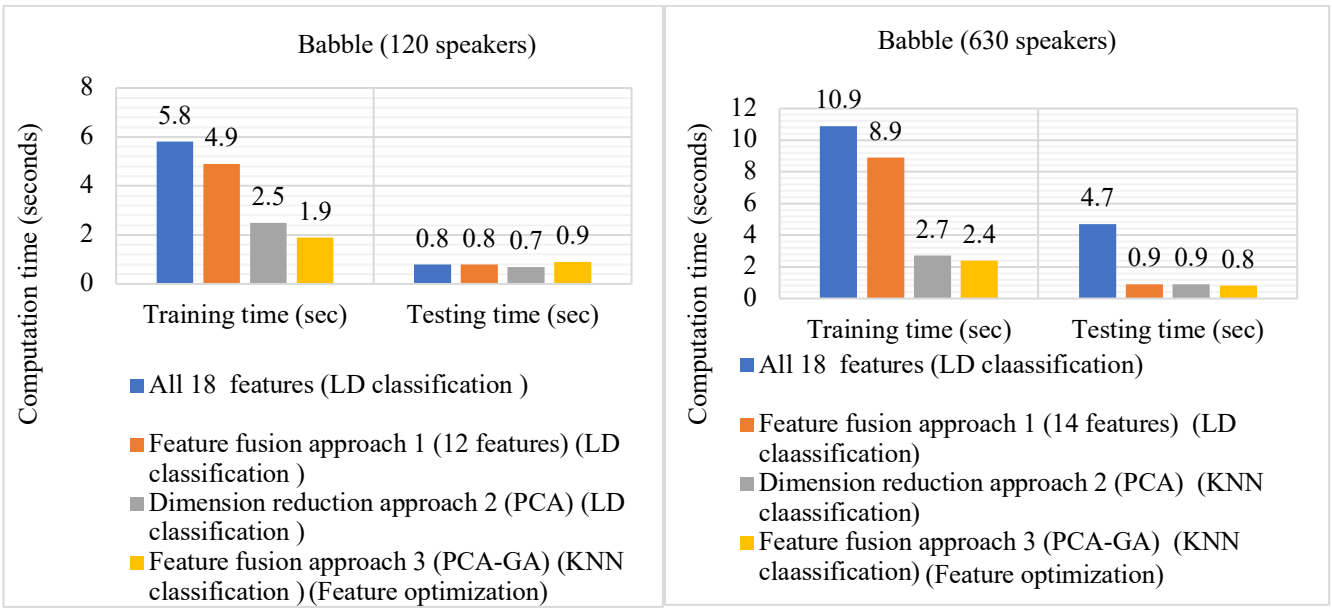


Fig.41 Computation timing of best models for babble noise data.

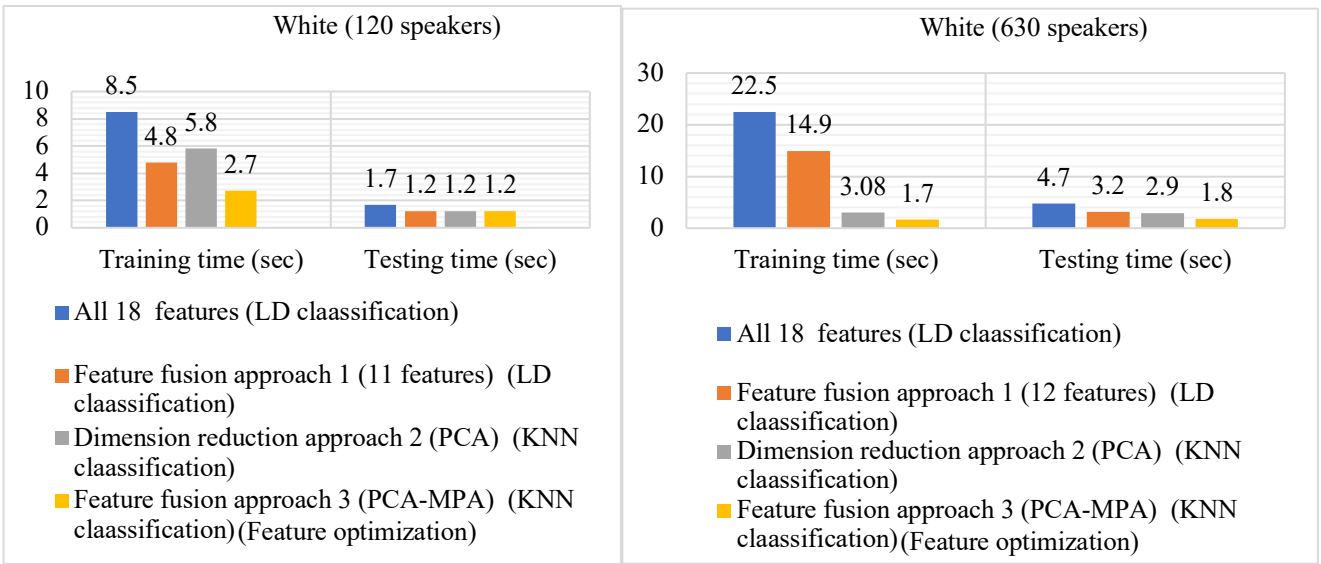


Fig.42 Computation timing of best models for white noise data.

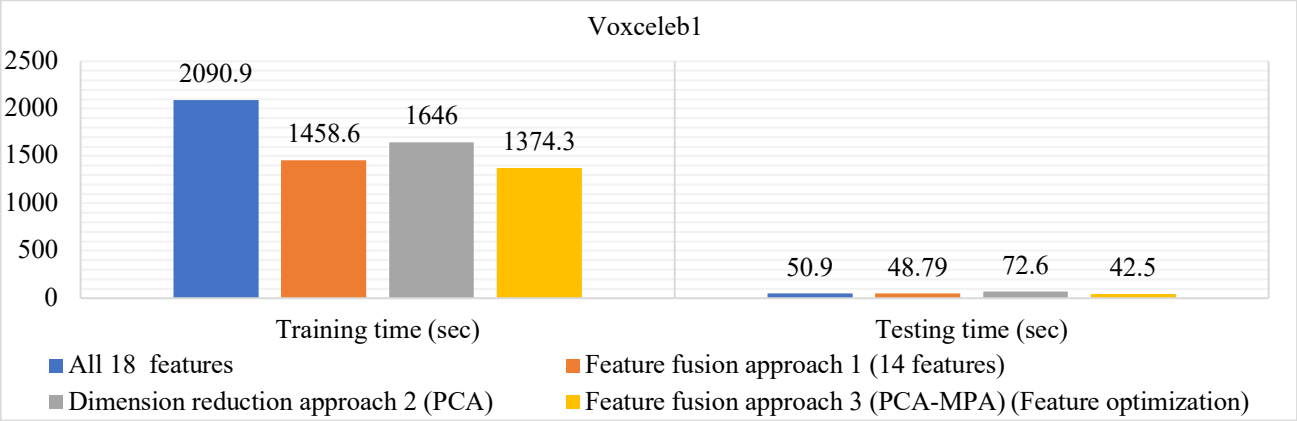


Fig.43 Computation timing of best models for voxceleb1.

Chapter 5

Conclusion

5.1 Result observation

Following points describes factors that affect SR performance

1. Feature Fusion: Larger feature fusions do not always lead to better SR performance. In some cases, models with smaller feature fusions outperform those with more features. This suggests that careful selection and combination of features are crucial for optimal results.
2. Feature Optimization: Among the three proposed approaches, Feature Optimization with PCA-GA, and PCA-MPA consistently delivers better results in most cases. Notably, it significantly reduces computation timing, making it a promising technique for improving efficiency.
3. Impact on Classification: The choice of approach affects the performance of different classifiers. KNN classification benefits from Dimension Reduction and Feature Optimization, while LD and Ensemble classifiers perform better with Feature Level Fusion.
4. Dataset Influence: The input dataset plays a significant role in SR performance. For TIMIT babble noise data, Feature Level Fusion and PCA-GA feature optimization demonstrate superior results, while TIMIT White noise data benefits from PCA dimension reduction and PCA-MPA feature optimization. PCA-MPA also performs well for the voxceleb1 dataset.

These findings highlight the importance of tailoring the approach to the specific dataset and classifier, and they provide valuable insights for designing effective SR systems.

5.2 Limitations

While designing a speaker recognition system, the following limitations should be considered:

1. Emotional variability affects accuracy.
2. Noise interference degrades performance.
3. Vulnerable to voice mimicry and spoofing.
4. Illness and aging alter voice.
5. Challenges with language and accent diversity.
6. High computational and storage demands.

5.3 Conclusion

The first part focuses on a novel fusion technique that combines 18 different speech features, including mel frequency cepstral coefficient (MFCC), linear predictive coding (LPC), perceptual linear prediction (PLP), root mean square (RMS), centroid, and entropy features. By incorporating their respective delta (Δ) and delta–delta (ΔΔ) feature vectors, this approach demonstrates a substantial enhancement in speaker identification accuracy and a notable reduction in equal error rates (EER) for speaker verification across diverse datasets. The experimental results show that the SI accuracy of the system increases to 100% and the EER value is reduced to 0% when multiple fusions of features are tested on ELSDSR, voxforge, and VCTK data. For the NIST-2008 dataset, the proposed model achieves the best SI accuracy of 96.9% with the fusion of 11, 12 and 14 features and the best EER of 0.2% with the fusion of 11 features using the LD classifer. For voxceleb1, the fusion of 14 and 15 features gave the best SI accuracy of 90% and 89.3% and SV EER values of 4.07% and 4.31%, respectively. From the experimental results, it is observed that the fusion of PLP, LPC, PLP, ΔΔLPC, RMS, MFCC, ΔΔRMS, ΔΔPLP, ΔΔentropy, ΔLPC, entropy, Δentropy, ΔMFCC, and ΔRMS (14 features) gives the best SI and SV results on all fve speech datasets, from which it can be concluded that the proposed model with the fusion of 14 features is suitable for various sizes of speech datasets.

The second part of the research delves into a comprehensive study on speaker recognition, emphasizing three pivotal strategies: feature-level fusion, dimension reduction using principal component analysis (PCA) and independent component analysis (ICA), and feature optimization employing genetic algorithm (GA) and marine predators algorithm (MPA). By

implementing a combination of PCA with GA and MPA, this approach significantly enhances the performance of speaker recognition systems, particularly evident in various datasets with diverse noise levels and speaker counts.
Notably, TIMIT babble noise dataset (120 speakers) achieved a speaker identification accuracy of 92.7% using feature fusion and a speaker verification equal error rate (SV EER) of 0.7% with various feature optimization techniques (PCA-GA) and LD and KNN classifiers. For the TIMIT babble noise dataset (630 speakers), a speaker identification accuracy of 93.5% and SV EER of 0.13% were obtained using KNN classifiers with feature optimization. Similarly, the TIMIT white noise dataset (120 and 630 speakers) achieved speaker identification accuracies of 93.3% and 83.5%, and SV EER values of 0.58% and 0.13% respectively, utilizing PCA dimension reduction and feature optimization techniques (PCA-MPA) with KNN classifiers. Moreover, the voxceleb1 dataset achieved a speaker identification accuracy of 95.2% and SV EER of 1.8% through PCA-MPA feature optimization with KNN classifiers.

Approach 3, which combines principal component analysis (PCA) with genetic algorithm (GA) and marine predators algorithm (MPA), consistently outperforms all other approaches for both noisy and multimedia datasets. The superior performance of this method underscores its effectiveness in handling various data types and noise levels.

5.4 Future work

In the future, we can address the limitations of speaker recognition by:

1. Improving feature capture to better recognize emotional states.
2. Combining text and audio data to build models using Large Language Models (LLMs).
3. Focusing on model optimization techniques to handle large audio datasets effectively.
4. Implementing feature optimization alongside model optimization to enhance overall performance.

Publication

Journal Papers

- **Chauhan Neha** & Isshiki Tsuyoshi & Li Dongju. (2023). Text-Independent Speaker Recognition System Using Feature-Level Fusion for Audio Databases of Various Sizes. SN Computer Science. 4. 10.1007/s42979-023- 02056-w.
- **Chauhan N**.; Isshiki, T.; Li, D. Enhancing Speaker Recognition Models with Noise-Resilient Feature Optimization Strategies. *Acoustics* 2024, *6*, 439-469. https://doi.org/10.3390/acoustics6020024

International Conference Papers

- **Neha Chauhan**, T. Isshiki and D. Li, "Speaker Recognition Using LPC, MFCC, ZCR Features with ANN and SVM Classifier for Large Input Database," *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, Singapore, 2019, pp. 130-133, DOI: 10.1109/CCOMS.2019.8821751.Citation: 41
- **Neha Chauhan,** T. Isshiki and D. Li, "Speaker Recognition using fusion of features with Feedforward Artificial Neural Network and Support Vector Machine," *2020 International Conference on Intelligent Engineering and Management (ICIEM)*, London, UK, 2020, pp. 170-176, DOI: 10.1109/ICIEM48762.2020.9160269.Citation: 6

Reference

1. Picheny M; Nahamou D, Goel V, Kingbusy B, Ramabhadran S.J Saon, G “Trends and Advances in Speech recognition” IBM Journal of Research and Development, Vol no-5 PP-2:1-2:18 sept-oct-2011.
2. L.R. Rabiner, “A tutorial on Hidden Markov Models and selected applications in speech recognition”, Proc. IEEE, 77(2), 1989, 257-286.
3. L.R Rabiner J. G. Wilpon, “Speaker independent isolated word recognition for a moderate size voculabary”, IEEE Transaction on Acoustics
4. Chauhan Neha & Isshiki Tsuyoshi & Li Dongju. (2023). Text-Independent Speaker Recognition System Using Feature-Level Fusion for Audio Databases of Various Sizes. SN Computer Science. 4. 10.1007/s42979-023-02056-w.
5. Lu, X., Dang, J. Dimension reduction for speaker identification based on mutual information. Proc. Interspeech; 2007.pp. 2021-2024. https://doi.org/10.21437/Interspeech.2007-165
6. M. Zamalloa, G. Bordel, L. J. Rodriguez and M. Penagarikano, "Feature Selection Based on Genetic Algorithms for Speaker Recognition," *2006 IEEE Odyssey - The Speaker and Language Recognition Workshop*, San Juan, PR, USA, 2006, pp. 1-8, doi: 10.1109/ODYSSEY.2006.248087.
7. D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.
8. Rai, R., Dhal, K.G., Das, A. *et al.* An Inclusive Survey on Marine Predators Algorithm: Variants and Applications. *Arch Computat Methods Eng* 30, 3133–3172 (2023). https://doi.org/10.1007/s11831-023-09897-x
9. Salama, Dr-Diaa & Nabil, Ayman & Ibrahim, Shimaa & Houssein, Essam. (2021). An Efficient Marine Predators Algorithm for Feature Selection. IEEE Access. 9. 1-18. 10.1109/ACCESS.2021.3073261.
10. El-Samie FEA. Information security for automatic speaker identification. Springerbriefs in electrical and computer engineering. Berlin: Springer; 2011.
11. Barbu T. A supervised text-independent speaker recognition approach. Int J Electron Commun Eng. 2007;1:2726–30.
12. de Lara JRC. A method of automatic speaker recognition using cepstral features and vectorial quantization. In: Sanfeliu A, Cortés ML, editors. Progress in pattern recognition, image analysis and applications. CIARP 2005. Lecture notes in computer science. Berlin, Heidelberg: Springer; 2005. pp. 146–53.
13. Minh ND. An automatic speaker recognition system. Lausanne, Switzerland: Audio Visual Communications Laboratory Swiss Federal Institute of Technology; 1996.
14. Lei HH. Structured approaches to data selection for speaker recognition. Technical Report No. UCB/EECS. Berkeley: University of California; 2010.
15. Chaudhary R. Short-term spectral feature extraction and their fusion in text independent speaker recognition: a review. BIJIT BVICAM’s Int J Inf Technol. 2013;5:630–9.
16. Furui S. Comparison of speaker recognition methods using statistical features and dynamic features. IEEE Trans Acoust Speech Signal Process. 1981;29:342–50. https://doi.org/10.1109/TASSP.1981.1163605
17. Kermorvant C, Morris A. A comparison of two strategies for ASR in additive noise: missing data and spectral subtraction. In: Proceedings of the 6th European conference on speech communication and technology. 1999. pp. 2841–4.
18. Varga AP, Moore RK. Hidden Markov model decomposition of speech and noise. In: International conference on acoustics, speech, and signal processing. Albuquerque, NM, USA: IEEE; 1990. pp. 845–8 vol.2.
19. Mittal U, Phamdo N. Signal/noise KLT based approach for enhancing speech degraded by colored noise. In: Proceedings of IEEE international conference on acoustics, speech, and signal processing. (Cat. No.00CH37100). Istanbul, Turkey: IEEE; 2000. pp. 1847–50.
20. Hu Y, Loizou PC. Subjective comparison and evaluation of speech enhancement algorithms. Speech Commun. 2007;49:588–601. https://doi.org/org/10.1016/j.specom.2006.12.006
21. Vaseghi SV, Milner BP. Noise compensation methods for hidden Markov model speech recognition in adverse environments. IEEE Trans Speech Audio Process. 1997;5:11–21. https://doi.org/10.1109/89.554264
22. Boll S. Suppression of acoustic noise in speech using spectral subtraction. IEEE Trans Acoust Speech Signal Process. 1979;27:113–20. https://doi.org/10.1109/tassp.1979.116320.
23. Hermansky H, Morgan N. RASTA processing of speech. IEEE Trans Speech Audio Process. 1994;2:578–89. https://doi.org/10.1109/89.326616

24. Hermansky H, Morgan N, Bayya A, Kohn P. Compensation for the effect of the communication channel in auditory-like analysis of speech (RASTAPLP). In: Proceedings of 2nd European conference on speech communication and technology (Eurospeech 1991). Genovo, Italy; 1991. pp. 1367–70.
25. Thyme-Gobbel AE, Hutchins SE. On using prosodic cues in automatic language identification. In: Proceeding of fourth international conference on spoken language processing. Philadelphia, PA, USA: IEEE; 1996. pp. 1768–71.
26. Mary L, Yegnanarayana B. Extraction and representation of prosodic features for language and speaker recognition. Speech Commun. 2008;50:782–96. https://doi.org/10.1016/j.specom.2008.04.010
27. Kumari TRJ, Jayanna HS. Limited data speaker verification: fusion of features. Int J Electr Comput Eng. 2017;7:3344–57. https://doi.org/10.11591/ijece.v7i6
28. Chauhan N, Isshiki T, Li D. Speaker recognition using fusion of features with feedforward artificial neural network and support vector machine. In: International conference on intelligent engineering and management (ICIEM). London, UK: IEEE; 2020. pp. 170–6.
29. Adami AG, Mihaescu R, Reynolds DA, Godfrey JJ. Modeling prosodic dynamics for speaker recognition. In: Proceedings of 2003 IEEE international conference on acoustics, speech, and signal processing. Hong Kong, China: IEEE; 2003. pp. IV–788.
30. Hossan MA, Memon S, Gregory MA. A novel approach for MFCC feature extraction. In: 4th international conference on signal processing and communication systems. Gold Coast, QLD, Australia: IEEE; 2011. pp. 1–5.
31. Peacocke RD, Graf DH. An introduction to speech and speaker recognition. Computer. 1990;23:26–33. https://doi.org/10.1109/2.56868
32. Kumar K, Kim C, Stern RM. Delta-spectral cepstral coefficients for robust speech recognition. In: IEEE international conference on acoustics, speech and signal processing. Prague, Czech Republic: IEEE; 2011. pp. 4784–7.
33. Sönmez MK, Shriberg E, Heck LP, Weintraub M. Modeling dynamic prosodic variation for speaker verification. In: The 5th international conference on spoken language processing. Sydney, Australia: Sydney Convention Centroid; 1998. pp. 3189–9192.
34. Carey MJ, Parris ES, Lloyd-Thomas H, Bennett S. Robust prosodic features for speaker identification. In: Proceeding of fourth international conference on spoken language processing. Philadelphia, PA, USA: IEEE; 1996. pp. 1800–3.
35. Chauhan N, Isshiki T, Li D. Speaker recognition using LPC, MFCC, ZCR features with ANN and SVM classifier for large input database. In: IEEE 4th international conference on computer and communication systems (ICCCS). Singapore: IEEE; 2019. pp. 130–3.
36. Lip CC, Ramli DA. Comparative study on feature, score and decision level fusion schemes for robust multibiometric systems. In: Sambath S, Zhu E, editors. Frontiers in computer education. Berlin, Heidelberg: Springer; 2012. pp. 941–8.
37. Alam MJ, Kenny P, Stafylakis T. Combining amplitude and phase-based features for speaker verification with short duration utterances. In: Proceedings of the 16th annual conference of the international speech communication association. Interspeech. Dresden, Germany: 2015. pp. 249–53.
38. Li Z, He L, Zhang W, Liu J. Multi-feature combination for speaker recognition. In: 7th international symposium on Chinese spoken language processing. Tainan, Taiwan: IEEE; 2010. pp. 318–21.
39. Hosseinzadeh D, Krishnan S. Combining vocal source and MFCC features for enhanced speaker recognition performance using GMMs. In: IEEE 9th workshop on multimedia signal processing. Chania, Greece: IEEE; 2007. pp. 365–8.
40. Nakagawa S, Wang L, Ohtsuka S. Speaker identification and verification by combining MFCC and phase information. IEEE Trans Audio Speech Lang Process. 2012;20:1085–95. https://doi.org/10.1109/tasl.2011.2172422
41. Venturini A, Zao L, Coelho R. On speech features fusion, α-integration Gaussian modeling and multi-style training for noise robust speaker classification. IEEE/ACM Trans Audio Speech Lang Process. 2014;22:1951–64. https://doi.org/10.1109/taslp.2014.2355821
42. Elmir Y, Elberrichi Z, Adjoudj R. Score level fusion based multimodal biometric identification (Fingerprint & voice). In: 6th international conference on sciences of electronics, technologies of information and telecommunications (SETIT). Sousse, Tunisia: IEEE; 2012. pp. 146–50.
43. Velayuthapandian, Karthikeyan & S, Suja. Hybrid machine learning classification scheme for speaker identification. Journal of Forensic Sciences,2022. 67. 10.1111/1556-4029.15006.
44. Banerjee A, Dubey A, Menon A, Nanda S, Nandi GC. Speaker recognition using deep belief networks. arXiv:1805.08865. 2019.
45. Gupta M, Bharti SS, Agarwal S. Gender-based speaker recognition from speech signals using GMM model. Mod Phys Lett B. 2019;33:1950438. https://doi.org/10.1142/s0217984919504384
46. Assaad FS, Serpen G. Transformation based score fusion algorithm for multi-modal biometric user authentication through ensemble classification. Procedia Comput Sci. 2015;61:410–5. https://doi.org/10.1016/j.procs.2015.09.175
47. Dehak N, Kenny PJ, Dehak R, Dumouchel P, Ouellet P. Front-end factor analysis for speaker verification. IEEE Trans Audio Speech Lang Process. 2011;19:788–98.

https://doi.org/10.1109/tasl.2010.2064307
48. Dhakal P, Damacharla P, Javaid A, Devabhaktuni V. A near real-time automatic speaker recognition architecture for voice-based user interface. Mach Learn Knowl Extr. 2019;1:504–20. https://doi.org/10.3390/make1010031
49. Medikonda J, Bhardwaj S, Madasu H. An information set-based robust text-independent speaker authentication. Soft Comput. 2019;24:5271–87. https://doi.org/10.1007/s00500-019-04277-9
50. Soltane CB, Kelbesa IY. An Intelligent text independent speaker identification using VQ-GMM model based multiple classifier system. World Acad Sci Eng Technol Int J Comput Inf Eng. 2010;8:1949–58.
51. Ahmad KS, Thosar AS, Nirmal JH, Pande VS. A unique approach in text independent speaker recognition using MFCC feature sets and probabilistic neural network. In: Eighth international conference on advances in pattern recognition. Kolkata, India: IEEE; 2015. pp. 1–6.
52. Bhardwaj S, Srivastava S, Hanmandlu M, Gupta JRP. GFM-based methods for speaker identification. IEEE Trans Cybern. 2013;43:1047–58. https://doi.org/10.1109/TSMCB.2012.2223461
53. Al-Kaltakchi MTDS, Woo WL, Dlay S, Chambers JA. Evaluation of a speaker identification system with and without fusion using three databases in the presence of noise and handset effects. EURASIP J Adv Signal Process. 2017;2017:1–17. https://doi.org/10.1186/s13634-017-0515-7
54. Al-Kaltakchi MTS, Woo WL, Dlay SS, Chambers JA. Comparison of I-vector and GMM-UBM approaches to speaker identification with TIMIT and NIST 2008 databases in challenging environments. In: 25th European signal processing conference (EUSIPCO). Kos, Greece: IEEE; 2017. pp. 533–7.
55. Okabe K, Koshinaka T, Shinoda K. Attentive statistics pooling for deep speaker embedding. arXiv:1803.10963. 2018.
56. Nagrani A, Chung JS, Zisserman A. Voxceleb: a large-scale speaker identification dataset. arXiv:1706.08612. 2017.
57. Loughran, Roisin & Agapitos, Alexandros & Kattan, Ahmed & Brabazon, Anthony & O'Neill, Michael. (2017). Feature selection for speaker verification using genetic programming. Evolutionary Intelligence. 10. 10.1007/s12065-016-0150-5.
58. Al-Kaltakchi MTDS, Woo WL, Dlay S, Chambers JA. Evaluation of a speaker identification system with and without fusion using three databases in the presence of noise and handset effects. EURASIP J Adv Signal Process. 2017;2017:1–17. https://doi.org/10.1186/s13634-017-0515-7
59. Al-Kaltakchi MTS, Woo WL, Dlay SS, Chambers JA. Comparison of I-vector and GMM-UBM approaches to speaker identification with TIMIT and NIST 2008 databases in challenging environments. In: 25th European signal processing conference (EUSIPCO). Kos, Greece: IEEE; 2017. pp. 533–537. https://doi.org/10.23919/EUSIPCO.2017.8081264
60. Zou, Xin & Jancovic, Peter & Liu, Ju.The effectiveness of ICA-based representation: Application to speech feature extraction for noise robust speaker recognition;2006. pp.1-5.
61. Mohammadi, Mohsen & Sadegh Mohammadi, H. R. Study of speech features robustness for speaker verification application in noisy environments;2016. pp.489-493. 10.1109/ISTEL.2016.7881869
62. F. Meriem, H. Farid, B. Messaoud and A. Abderrahmene. Robust Speaker Verification Using a New Front End Based on Multitaper and Gammatone Filters.2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems; 2014. pp. 99-103. https://doi.org/10.1109/SITIS.2014.111
63. Lartillot O, Toiviainen P. MIR in Matlab (II): a toolbox for musical feature extraction from audio. In: Proceedings of the 10th international conference on digital audio effects. Bordeaux, France. 2017. pp. 127–30.
64. Wang L, Chen Z, Yin F. A novel hierarchical decomposition vector quantization method for high-order LPC parameters. IEEE/ACM Trans Audio Speech Lang Process. 2015; 23:212–21. https://doi.org/10.1109/TASLP.2014.2380352
65. Slifka J, Anderson TR. Speaker modification with LPC pole analysis. In: International conference on acoustics, speech, and signal processing. Detroit, MI, USA: IEEE; 1995. pp. 644–7.
66. Daniel PW. PLP, RASTA, MFCC and inversion in Matlab. 2005.@misc{Ellis05-rastamat; http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/.
67. Hermansky H. Perceptual linear predictive (PLP) analysis of speech. J Acoust Soc Am. 1990;87:1738–52. https://doi.org/10.1121/1.399423
68. Chauhan N, Chandra M. Speaker recognition and verification using artificial neural network. In: Conference on wireless communications, signal processing and networking (WiSPNET). Chennai, India: IEEE; 2017. pp. 1147–9. https://doi.org/10.1109/WiSPNET.2017.8299943
69. Ross A. Fusion, feature-level. In: Li SZ, Jain A, editors. Encyclopedia of biometrics. Boston, MA: Springer; 2009. pp. 597–602.
70. Root-mean-square value. A Dictionary of Physics (6 ed.). Oxford University Press. 2009; ISBN 9780199233991.
71. You, S.D.; Hung, M.-J. Comparative Study of Dimensionality Reduction Techniques for Spectral–Temporal Data. Information 2021, 12, 1. https://doi.org/10.3390/info12010001
72. https://medium.com/analytics-vidhya/understanding-principle-component-analysis-pca-step-by-step-e7a4bb4031d9

73. J. Herault, C. Jutten, B. Ans, Detection de grandeurs primitives dans un message composite par une architecture de calcul neuromimetique en apprentissage non supervise. In: 10 Colloque sur le traitement du signal et des images, FRA, 1985.GRETSI, Groupe d'Etudes du Traitement du Signal et des Images 1985.
74. Tharwat, Alaa. Independent Component Analysis: an Introduction. Applied Computing and Informatics.2018. ahead-of-print. 10.1016/j.aci.2018.08.006.
75. Zhao Y, Sun P-P, Tan F-L, Hou X and Zhu C-Z. NIRS-ICA: A MATLAB Toolbox for Independent Component Analysis Applied in fNIRS Studies. Front. Neuroinform.2021. 15:683735. doi: 10.3389/fninf.2021.683735
76. Wang, Aiguo & An, Ning & Chen, Guilin & Li, Lian & Alterovitz, Gil. (2015). Accelerating wrapper-based feature selection with K-nearest-neighbor. Knowledge-Based Systems. 83. 10.1016/j.knosys.2015.03.009.
77. Abdulhamit Subasi. Practical Machine Learning for Data Analysis Using Python. Machine learning techniques ISBN 9780128213797;2020. pp. 91-202. https://doi.org/10.1016/B978-0-12-821379-7.00003-5.
78. Yao Z, Ruzzo WL. A Regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data. BMC Bioinform. 2006;7:S11. https://doi.org/10.1186/1471-2105-7-S1-S11
79. Dietterich TG. Ensemble learning. In: Arbib MA editor. The handbook of brain theory and neural networks. Cambridge, MA, USA: MIT Press; 2012. pp. 110–25.
80. Tin Kam H. The random subspace method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell. 1998;20:832–44. https://doi.org/10.1109/34.709601
81. Feng L. Speaker recognition, informatics and mathematical modelling. Denmark: Technical University of Denmark; 2004.
82. NIST Multimodal Information Group. NIST speaker recognition evaluation test set LDC2011S08. Web download. Philadelphia: Linguistic Data Consortium; 2008.
83. Release notes 2.4.2. Audacity Wiki. 2020.
84. Abdulaziz, Azhar, and Veton Kepuska. Noisy TIMIT Speech LDC2017S04. Web Download. Philadelphia: Linguistic Data Consortium, 2017. https://doi.org/10.35111/m440-jj35
85. Saito T, Rehmsmeier M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS One. 2015;10:e0118432. https://doi.org/10.1371/journal.pone.0118432
86. Tharwat A. Classification assessment methods: a detailed tutorial. Appl Comput Inform. 2020; 17:168–92. https://doi.org/10.1016/j.aci.2018.08.003
87. N. N. Prachi, F. M. Nahiyan, M. Habibullah and R. Khan, "Deep Learning Based Speaker Recognition System with CNN and LSTM Techniques," 2022 Interdisciplinary Research in Technology and Management (IRTM), Kolkata, India, 2022, pp. 1-6, doi: 10.1109/IRTM54583.2022.9791766.
88. Mandalapu, Hareesh & Ramachandra, Raghavendra & Busch, Christoph. (2020). Multilingual Voice Impersonation Dataset and Evaluation.
89. Cai, Weicheng, Jinkun Chen, and Ming Li. "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system." *arXiv preprint arXiv:1804.05160* (2018).
90. Jakubec, Maros & Jarina, Roman & Lieskovska, Eva & Kasák, Peter. (2023). Deep speaker embeddings for Speaker Verification: Review and experimental comparison. Engineering Applications of Artificial Intelligence,2023. 10.1016/j.engappai.2023.107232.