Robustness of features and models for text-dependent speaker verification

Dhaivat Joshi[1], *Janaki Sheth*[2], *Vijay Ravi*[1], *Zhi Ming Chua*[1]

[1]Dept. of Electrical Engineering, University of California, Los Angeles
[2]Dept. of Physics, University of California, Los Angeles

djjoshi@ucla.edu, janaki.sheth@physics.ucla.edu, vijaysumaravi@ucla.edu, zmchua@ucla.edu

Abstract

Speaker verification is an integral aspect of speaker recognition, wherein test speech samples are determined to come from the same or different speakers. There are several ensuing challenges such as the presence of background noise and short lengths of utterances, that make verification a difficult task.

In this paper, we probed this question using a dataset comprising of speech samples from 50 male speakers with 236 unique signals for each of the clean and noisy training datasets. Noise for this project comprises of babble noise in the background. A set of two signals from the same speaker are labeled 1 and the others are labeled 0. We use data sampling to balance the resulting highly unbalanced training set.

Feature vectors for voiced frames of the speech signals were estimated using Voicebox and the Matlab signal processing toolbox. Using a noise detector to determine if the two input signals were clean or noisy, the signals were passed through either the clean or noisy model respectively. The models comprise of four classifiers each and use a score fusion technique to give a final prediction.

We further also probe which classifiers have better performance under the clean and noisy conditions, and which features exhibit robustness. Our results indicate that the model trained on clean dataset beats the given baseline for the corresponding test set while resulting in a high error rate for the dissimilar test set. Similar result holds true for the babble noise trained model, while the multi dataset trained algorithm shows significant improvement on both test datasets. This is suggestive of the need for a noise detector in combination with the models. Further analysis, show GFCCs to be noise robust and improve classifier performance under both conditions.

Index Terms: speech verification, noise robustness, human-computer interaction

1. Introduction

The goal of speaker verification is to discern the likelihood of the utterance of the test speech samples as being from the same speaker or two different speakers. Intra-speaker variability based on mood, intonation, speed and inter-speaker variability make it a difficult task to accurately distinguish speakers, especially in noisy environments. Thus, success in this task depends on extracting speaker-dependent features that are robust in clean and noisy environments, and models that accurately capture the difference between two speakers.

In this manuscript, we explore this problem in clean and babble noise conditions. We are further constrained to a known sentence i.e. text-dependent conditions. Additionally, we break the problem down into two parts - training on clean data, and training on noisy data, where each trained model is then tested on both clean and noisy data. Lastly, we also computed the performance of training on a multi dataset, which consists of both clean and noisy data, and testing on a separate multi dataset. This leads to a preliminary insight into which features and models are robust and which ones exhibit a significant change in performance.

2. Literature Review

Speaker recognition has been a burgeoning field, with research activity from several universities and labs [1]. It involves identification - identifying the speaker given the test speech sample, and verification - differentiating the speakers of two test samples. Hence, in this manuscript we use multiple features and models previously used in the context of automatic speaker recognition for the assigned task of speaker verification.

Feature extraction retrieves speaker-specific properties from the raw signal and converts them to numerical vectors that further feed into models; the results of which can be compared to study the likelihood of the occurrence of the test data from the same speaker.

Kinnunen [2] suggests choosing features which ensure greater inter-speaker variability, and less intra-speaker variability, as well as those that are robust in noisy conditions. They can be further categorized into (1) short-term spectral features, (2) voice source features, (3) spectro-temporal features, (4) prosodic features and (5) high-level features.

As speech is a continuously changing signal, the analysis comprises of extracting features for short frames of 10-20 ms duration, often requiring the use of a suitable window to prevent edge effects of STFT. [3] It has been found that F0, H1-H2 (amplitude difference between the first and second harmonics), H2-H4 (the slope of the spectrum from second to the fourth) and H4-2KHz are illustrative of the voice source of the speaker. [4] Meanwhile, formants relay vocal tract information. As discussed in [5] F1, F2 vary considerably when a vowel is articulated while F3 could be used to ascertain vocal tract length. These can also be derived using linear predictive coding (LPC) [7] on assuming the all-pole model of speech production.

Motivated by psycho-acoustic studies of how we perceive sound, features such as mel-frequency cepstral coefficients (MFCC) [David 1980], power-normalized cepstral coefficients (PNCC) [8], perceptual linear prediction (PLP) [9] and gammatone filterbank cepstral coefficients (GFCC) [10] were introduced. Usually representing the lower frequency range with higher resolution, MFCCs are computed by mapping the Hz frequency scale onto the mel scale using a triangular filterbank. Further, representative of the physiological cochlear filters, those used for GFCCs are described by the Equivalent Rectangular Bandwidth [11]. PLP was proposed as a solution for the inability of the LPC analysis to preserve or discard the spectral details of speech according to their auditory relevance [9].

In text-dependent recognition, the presence of similar phoneme sequences enables us to align the test and train sequences and the derived features. Dynamic Time Warping has been recognized in previous literature [2] as a method to directly compare these vectors. Additionally, stochastic models such as the Hidden Markov model (HMM) have also been extensively studied. They are used to model the speaker features using a hidden probability distribution whose parameters are determined using the training data set. The test results are evaluated using the likelihood of the test utterance with respect to these hidden Markov states. Analogously, in text-independent recognition, Gaussian Mixture Models (GMM) are found effective for short-term feature modeling [15].

Further, Kinnunen [2] details that classifiers such as support vector machines (SVMs) and neural networks are used to describe the boundary between the two speakers. Thus, they seem well-suited for speaker verification.

Using multiple models for speaker recognition often demands the use of score-fusion [2] [15]. As is documented, the different classifiers could model same or different features, and are given equal or different weights optimized using cross-validation.

3. Features and Models

3.1. Feature extraction

Prior to extraction, the speech signal was segmented into sections of length 20 ms each. The set of features computed for each frame consists of pitch (F_0), the first four formants (F_1, F_2, F_3, F_4), MFCCs and their derivatives and double derivatives, LPC coefficients, PLP and GFCCs.

As previously alluded to, F_0 captures voice source information, and the formants and LPC describe vocal tract movement. MFCCs and GFCCs encapsulate both the source and tract information. The derivatives and double derivatives of the former computed using time differences, incorporate temporal information of the feature. Further, our features attempt to capture both spoken (pitch, formants and LPC) and perceived (MFCC, PLP, GFCC) speaker voice quality.

The pitch, formants, LPC and MFCCs were extracted using Voicebox [12]. PLP was extracted using the code developed by LabRosa in Columbia university. GFCC was extracted using code from the Perception and Neurodynamics Lab at OSU.

Eventually, we only retained the feature vector set for voiced sections of the training speech signals for comparison with the test data, since they were observed to be more robust to noise compared to the unvoiced sections.

3.2. Dataset Sampling

The dataset provided to us by the Speech Processing and Auditory Perception Lab, was an unbalanced set comprising of signals with both clean and babble noise backgrounds. These were renditions of the sentence, "Help the woman get back to her feet", spoken by 50 male speakers with each signal lasting for 1.5 to 2 seconds.

For each of the clean and noisy training dataset, the number of unique signals was 236, giving rise to ${}^{236}C_2 = 27,730$ pairs of speech signals. Each pair of speech signals is given a label, where label 1 indicates that both the speech samples of the example were spoken by the same speaker, and 0 indicates different speakers. There are a total of 586 pairs of speech samples with label 1, and 27,144 with label 0.

The imbalance in dataset results in poor performance for certain classifiers such as SVM. Hence, we re-sampled our dataset by oversampling the signal pairs labeled 1 by a factor of 2 to obtain $2 \times 586 = 1172$ data points with label 1, and undersampling those labeled 0 such that they are approximately 2.5 times the number of data points with label 1, which is around 2930. All our classifiers were trained on this newly created balanced dataset.

3.3. Models

The classifiers we feed the speaker-specific features to are: support vector machine (SVM), k-nearest neighbors (k-NN), naive Bayes (NB) and Gaussian Mixture Model (GMM).

SVM is a binary classifier that models the decision boundary between two classes by finding a hyperplane that maximizes the margin between them while penalizing misclassified data points. For the task of speaker differentiation, the training data set with speech samples from the same speaker is labeled by 1, and that with samples from different speakers is labeled 0. The classifier computes an optimal linear hyperplane in a higher dimensional kernel feature space, which is then used to label the testing data sets.

k-NN uses the "distance" between feature vectors of the test (centroid) and training data to classify the test sample. The closer the test sample is to a reference sample in the training set, the greater is the likelihood of it having the same label. For a regression model, the final result is derived using the weighted average of its k-nearest neighbors. This method requires further optimization of the number of nearest neighbors to suppress error caused by the presence of noise and prevent over-fitting.

NB uses similar assumptions as the Hidden Markov model, namely the conditional independence of the spectral feature vectors. The continuous nature of speech and correlation of subsequent frames, invalidates this assumption. Despite this, as shown in [13] HMMs perform well in the task of speaker recognition. Consequently, we further explored the use of the naive Bayes classifier with a kernel density probability distribution estimation in our project.

GMMs use an underlying probability distribution to model a speaker. This comprises of multiple Gaussians whose parameters μ, Σ are estimated by using the training data. We then use maximum log likelihood, as shown in [14], to determine the probability that the second signal in the pair is drawn from the distribution of the first signal. Similar to k-NN, the number of Gaussians were further optimized.

The Matlab machine learning toolbox was used for implementation of SVM, k-NN and NB. The implementation of GMMs was carried out using the codes in VoiceBox.

4. Results

4.1. Greedy Algorithm

The feature set of F_0, $F_1 - F_4$, and MFCCs with their single and double derivatives forms the core feature set. The other features were selected based on a greedy approach, where we incrementally test all our models by starting with the core features and adding the other features one-by-one to the feature set. This allows us to choose the best set of features for each of our models separately. One could also have used a global optimum strategy but it quickly becomes computationally costly as the number of features increases.

We trained on three different data sets - to determine the most optimal feature sets for each classifier under different noise conditions. These comprised of - clean set (Table 1),

Table 1: *Training on clean data set. Number of nearest neighbors used for k-NN is 5, and number of Gaussians for GMM is 4.*

					Clean test		Babble test	
	Core	GFCC	LPC	PLP	FPR	FNR	FPR	FNR
SVM	×	×	×		7.5	15.3	58.6	5.3
k-NN	×	×	×	×	13.8	9.4	79.5	1.3
NB	×				7.3	22.2	98.8	0.0
GMM	×				17.5	26.4	92.8	0.0

Table 2: *Training on babble noise data set. Number of nearest neighbors used for k-NN is 3, and number of Gaussians for GMM is 4.*

					Clean test		Babble test	
	Core	GFCC	LPC	PLP	FPR	FNR	FPR	FNR
SVM	×	×	×	×	33.1	61.2	20.2	27.7
k-NN	×	×	×		7.9	55.2	35.6	12.0
NB	×	×	×	×	0.0	100.0	21.4	21.3
GMM	×	×	×	×	0.0	99.4	30.1	23.3

babble noise set (Table 2) and a multi data set with both (Table 3). The greedy algorithm as applied to the training set determined the final set of features, which were later tested on the two testing sets - one with clean speech samples, the other with additive babble noise. The features incorporated per classifier are demarcated in the tables.

The results are documented as False Positive Rate (FPR) and False Negative Rate (FNR), which are given as percentages.

4.2. Score-Level Fusion

Since we used four different classifiers and different feature sets for each, we combined all of them using a technique called score-level fusion. Using linear regression, we optimized the fusion weights - weights given to each classifier - eventually giving a score for the two test sets for the models trained on the three different training data sets (Table 4).

Additionally, we were given baseline FPRs and FNRs derived using pitch of the signal as the sole discriminating feature, which are herewith compared to the results obtained.

5. Discussion

5.1. Performance of classifiers

As evident in Table 4, the combined classifier trained on the clean dataset beats the baseline for clean testing conditions, while it has a high FPR for the babble noise test set. This suggests that similar noise statistics in all the speech samples in this test set prompt the classifier to attribute most of the speech pairs to the same speaker, even when they originate from two different speakers.

The classifier trained on the noisy dataset performs considerably better on the corresponding test dataset compared to the clean set.

Lastly, the classifiers trained on the multi data set perform nearly equally well in the clean and babble noise conditions, beating the requisite baselines in both cases.

On further analysis of all three training cases, we observed that SVM has the lowest error rates when trained on the clean

Table 3: *Training on multi data set. Number of nearest neighbors used for k-NN is 3, and number of Gaussians for GMM is 8.*

					Clean test		Babble test	
	Core	GFCC	LPC	PLP	FPR	FNR	FPR	FNR
SVM	×	×	×	×	9.1	17.8	23.7	20.7
k-NN	×	×			12.8	17.0	40.6	13.2
NB	×			×	2.8	36.9	34.9	15.0
GMM	×	×	×		3.5	53.8	56.6	5.8

Table 4: *Score fusion using all 4 classifiers and respective features.*

		Testing Results		Baseline	
Training Set	Testing Set	FPR	FNR	FPR	FNR
Clean	Clean	5.9	12.6	34.0	30.0
	Babble	95.7	0.0	46.0	38.0
Babble	Clean	46.1	43.5	-	-
	Babble	22.2	19.4	-	-
Multi	Clean	4.5	30.2	32.0	33.0
	Babble	26.4	18.8	43.0	42.0

and multi data sets, while k-NN is the best classifier for the babble noise data set. Naive Bayes and GMM show considerably worse performance in the clean and babble noise training conditions. NB's performance might have worsened due to correlation of the spectral feature vectors for the speech signal set dissimilar to the training set. Also, the volatility of the GMM model when used for short-utterances is suggestive of the high error rates.

5.2. Noise robustness of features

To compare optimal features in the clean and noisy conditions, we additionally look at the SVM feature vectors (Fig. 1).

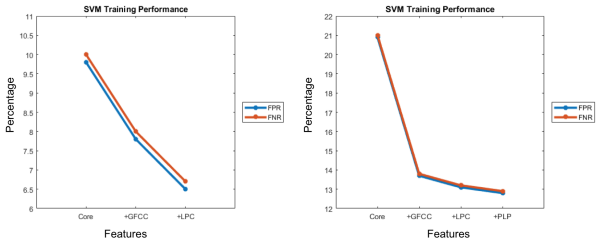

Figure 1: ***Training error rates of SVM classifier training:*** *Left: The training dataset contains clean speech samples. The greedy algorithm chooses Core($F_0, F_1 - F_4$, MFCC and derivatives) + GFCCs + LPC as the optimal features. Right: The training dataset comprises of the multi dataset i.e. mixture of clean and babble noise samples. Notice that the inclusion of GFCCs causes the greatest drop in training error rates in both conditions.*

GFCCs seem to cause the biggest incremental dip in training error rates when trained on the clean and multi datasets. This was further corroborated even on the babble noise set.

Similar analysis was conducted for the other classifiers,

wherein we compared the effect of using a feature vector set determined by only the Core features and that given by Core + GFCCs in both the clean and multi speech training dataset conditions. However, instead of the training errors, error percentages obtained by running the classifier on the test speech samples with clean and babble noise backgrounds are noted. The results for the multi training speech set are illustrated in Fig. 2.

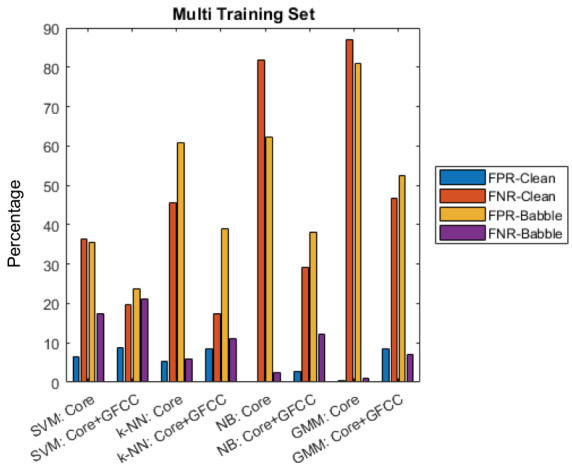

Figure 2: ***Comparison of Core + GFCC feature set vs Core feature set:*** *Across all the classifiers, it is evident that the addition of GFCCs to the feature set improves performance when the classifier is run on clean and babble noise test data.*

5.3. Feature Concatenation

We observed (Fig. 3) that on comparing the performances of the SVM classifier with different feature sets, when LPC was used as a feature in conjunction with GFCC, the performance improved although the performance worsens when LPC is the only additional feature to the core features. This suggests that feature concatenation may not be the best approach to combine features.

5.4. Final Algorithm

Comparing the scores in Table 4, we notice that the model trained on the clean dataset performs much better on the clean test set, and correspondingly for the one trained on the babble noise speech set. Meanwhile, the model trained using the multi data set produced FPR and FNR lower than the baseline for both the clean and noisy test sets.

This prompted us to consider a noise detector that acts as a switch determining the model that acts on the test dataset based on if noise is detected in the background. We design the noise detector to calculate the sum of the power spectral density of a frame of the speech signal. Then, the resulting vector of the length of total number of frames, is summed over to give a singular number per signal. Fig. 4 shows the power spectrum values for multiple frames of a single clean and babble noise signal, from which we deduce that the final number accorded to the noisy signal is larger than the clean signal. While, it might not always be the case that there exists a clear boundary between the clean and noisy samples, in our dataset we find there to be so.

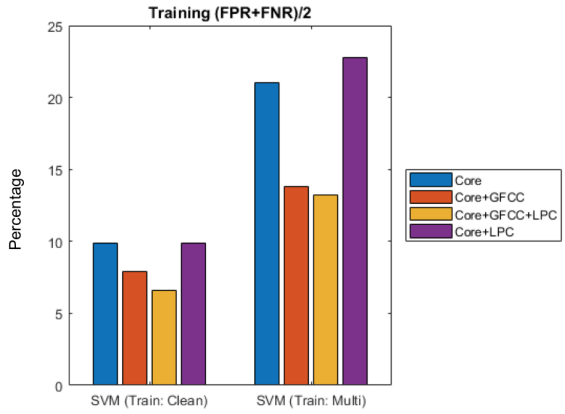

Figure 3: ***Comparison of GFCC and LPC:*** *In both clean and multi speech training cases, LPC further reduces training error rates when GFCC is part of the feature set, while the LPC worsens the performance when used in isolation.*

The clean speech signals gave a maximum value of ~ 900 a.u. and those with babble noise background gave a minimum value of ~ 1200 a.u. Thus, choosing a threshold of 1050, helps us segregate the test signals to choose which model is applied.

One could also consider filtering noise from the datasets. However, this would be easier if the background noise has values only at certain frequencies e.g. high frequency noise. The presence of babble noise which is made up of multiple simultaneous background speakers has a more distributed spread in its power spectrum. Hence, we resist from filtering out the noise and rather use two routes for the test dataset with models trained on clean and noisy signals respectively.

A second illustration in Fig. 5 compares frames from the two signals exhibiting the maximum power spectral density. The difference between the clean and noisy data is noticeable.

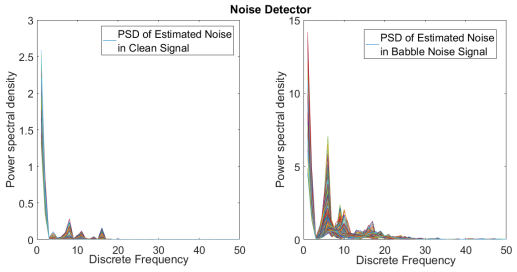

Figure 4: ***Power spectral density all frames for a sample clean and babble noise speech signal:*** *The noisy speech signal exhibits a larger power spectral density compared to the clean signal across nearly all frequencies. Thus, enabling us to create a threshold to differentiate the two.*

The noise detector is implemented using the `estnoiseg` function of the VoiceBox toolbox.

The final algorithm is illustrated in Fig. 6. It comprises of three components - the noise detector that classifies the incoming signals as noisy or clean, the respective models and the decision model that produces a final predicted label.

The noise detector directs the test signals to either the clean

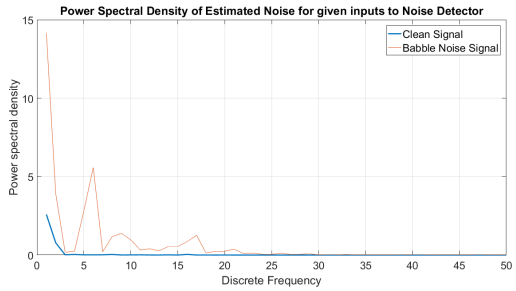

Figure 5: ***Frame with maximum power spectrum value of estimated noise for a sample clean and babble noise speech signal:*** *The frame from the clean speech signal shows lower power spectrum values of the estimated noise across all frequencies as compared to the frame with additive babble noise.*

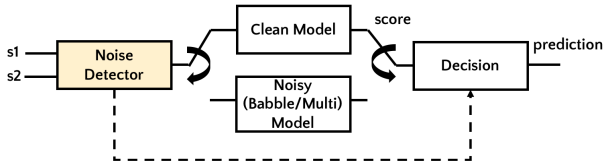

Figure 6: ***Final algorithm using the noise detector:*** *The two signals s_1, s_2 pass through the noise detector, corresponding model and the decision maker to give a final prediction of label 0 or 1.*

or noisy model. If both the signals are clean, the clean model is implemented. If either of them has babble noise background, the noisy model is used.

Amongst the two choices for the noisy model, the babble noise trained model has a lower equal error rate on the babble noise speech test set, than the multi trained model Table 4. However, the latter could be used for the cases when the noise detector is likely to make classification errors, as its performance on the clean dataset is superior to the former and also below the given baseline.

The results from the four classifiers of the respective model then undergo linear regression giving a final score of how likely are the two signals to arise from the same speaker or different speakers. The predicted label is determined based on the final score and a threshold learned during training.

The results for the final model is shown in Table 5.

Tabulating the error rates in Table 5, we observe that the choice of the babble speech trained model, gives lower values when implemented on the babble noise test set, as compared to the multi speech trained model. Further, testing on the multi dataset reveals a slight improvement by the babble speech model. However, one's choice of model is likely to be influenced by these results as well as the ability of the noise detector to correctly classify the speech samples.

6. Conclusions and Future Work

In this manuscript, we have described an algorithm comprising a noise detector, two models and score fusion that determines if the two speech test samples come from the same speaker or different speakers. The models comprise of four classifiers with

Table 5: ***FPR and FNR for final algorithm****:The clean + babble model combination slightly outperforms the clean + multi model combination for a multi testing set*

Models	Testing Set	Testing Results FPR	FNR
Clean + Babble	Clean	6.3	11.1
	Babble	22.5	19.8
	Multi	14.3	16.4
Clean + Multi	Clean	5.8	12.1
	Babble	28.3	19.1
	Multi	16.8	16.5

optimal feature sets determined using a greedy algorithm. For future work, we would like to probe these features by including derivatives and double derivatives. Also, LPCCs, PNCCs could be included to study their noise robustness in comparison to GFCCs. Further, we plan on implementing feature-level fusion to capture the difference between efficacy of fusion of classifiers and features. Lastly, we also like to use cross-validation to optimize the classifier hyperparameters.

7. Acknowledgements

We would like to thank Prof. Abeer Alwan, Gary Yeung and Soo Jin Park for their insightful discussions and guidance.

8. References

[1] J. P. Campbell, "Speaker recognition: a tutorial," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1437–1462, 1997.

[2] T . Kinnunen and H. Li "An overview of text-independent speaker recognition: From features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12-40, 2010.

[3] A. V. Oppenheim, and W. R. Schafer, *Discrete-Time Signal Processing*. Prentice Hall Press, 2009.

[4] J. Kreiman, S. J. Park, P. A. Keating and A. Alwan, "The Relationship Between Acoustic and Perceived Intraspeaker Variability in Voice Quality," in *Interspeech - In the sixteenth annual conference of the International Speech Communication Association*, 2015, pp. 2357–2360.

[5] C. Y. Epsy-Wilson, S. Manocha, and S. Vishnubhotla, "A new set of features for text-independent speaker identification," in *Proceedings of Interspeech*, 2006.

[7] J. Markhoul, "Linear Prediction: a tutorial review," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

[8] C. Kim and R. M. Stern, "Power-Normalized Cepstral Coefficients (PNCC) for Robust Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 7, pp. 1315–1329, 2016.

[9] H. Hermansky, "Power-Normalized Cepstral Coefficients (PNCC) for Robust Speech Recognition," *J. Acoust. Soc. Am.*, vol. 87, no. 4, pp. 1738–1752, 1990.

[10] X. Zhao and D. Wang "Analyzing noise robustness of MFCC and GFCC features in speaker identification," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7204–7208, 2013.

[11] M. Moinuddin and A. N. Kanthi "Speaker Identification based on GFCC using GMM," *International Journal of Innovative Research in Advanced Engineering*, vol. 1, no. 8 pp. 224–232, 2014.

[12] M. Brookes "Voicebox: Speech processing toolbox for matlab.," *http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html*

[13] L. Toth, A. Kocsor, and J. Csirik "On Naive Bayes in Speech recognition," *Int. J. Appl. Math. Comput. Sci.*, vol. 15, no. 2, pp. 287–294, 2005.

[14] D. Reynolds "Speaker identification and verification using Gaussian mixture speaker models" *Speech Communication*, vol. 7, no. 1-2, pp. 91-108, 1995.

[15] J. H. L. Hansen and T. Hasan "Speaker Recognition by Machines and Humans: A tutorial review" *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74-99, 2015.