# SPEAKER RECOGNITION WITH MEL-FREQUENCY CEPSTRAL COEFFICIENTS

*Karthikeyan Lakshmana Doss*

The University of Texas at Dallas

## ABSTRACT

This paper proposes an open-set text-dependent speaker recognition system that is designed for small-scale voice recognition implementations. The system initially reads in audio from a speaker. The system then extracts sets of Mel-frequency cepstral coefficients from the audio and compares them to speaker-specific codebooks that were generated during training using vector quantization. The speaker with the lowest distortion is chosen and this associated distortion is compared with a fixed distortion threshold to filter out impostors. The speaker would be given access if their associated distortion is lower than this threshold value. The model yields an average accuracy of 96 % across all password phrases (digits from “0” to “9”) for closed-set recognition and an equal error rate of 2.2 % for open-set recognition.

***Index Terms***— Speaker recognition, Mel-frequency cepstral coefficients, vector quantization, Euclidean distance, LBG algorithm

## 1. INTRODUCTION

Security and privacy are two of the most important aspects of any system that houses user-specific information. User ID and password protection, face scanning, and hand printing are all notable security measures implemented for these data-sensitive systems [1,2]. Recently, another such security measure that has gained popularity is speaker (or voice) recognition. With speaker recognition, the computer is trained to identify a speaker based on features that are unique to their voice. Depending on their implementation, speaker recognition systems fall into two types: text-dependent and text-independent [3]. With text-dependent systems, the speaker can only be correctly identified if they speak a unique phrase (or password) every time. Here, the same phrase is spoken by the speaker to train and test the system. With text-independent systems, however, the speaker can be identified regardless of what they speak. For the purposes of recognizing a speaker based on a specific password phrase, a text-dependent model would yield better results as the set of features collected would hold information about the speaker’s voice characteristics and the phrase characteristics (how each phoneme in the phrase connects to its neighboring phonemes).

A speaker recognition system can be implemented for services used by members of a household. With this system, a home security system, for instance, could confirm the identity of a home member and prevent impostors from entering. An online service with a family account, as another example, could confirm the identity of the family member accessing their portal. A text-dependent speaker recognition model that can be used for these types of small-scale systems is proposed in this paper. The model assumes three things: the total number of speakers is small (the model here is tested with 6 speakers), the audio phrase is only a few seconds long (the model is tested with digits from “0” to “9”), and the speaker utters the same password every time for access (the model does not work for text-independent scenarios).

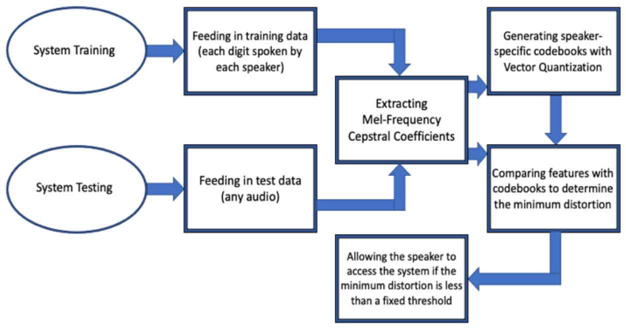

System Training
Feeding in training data (each digit spoken by each speaker)
Extracting Mel-Frequency Cepstral Coefficients
Generating speaker-specific codebooks with Vector Quantization
System Testing
Feeding in test data (any audio)
Comparing features with codebooks to determine the minimum distortion
Allowing the speaker to access the system if the minimum distortion is less than a fixed threshold

Fig. 1. The proposed speaker recognition model

To test that the proposed model can detect different users even if they have the same passwords, the model is trained and tested with the Free Spoken Digit Dataset (FSDD) [4]. This database consists of 6 unique speakers uttering the digits from “0” to “9” 50 times, with each utterance lasting for around 2 seconds. During the system training phase of the model, a single utterance of each digit by each speaker is fed into the system. From these audio files, a set of features called Mel-frequency cepstral coefficients are extracted and then trained with a technique called vector quantization, which generates “codebooks” that are unique to each speaker. During the system testing phase, any audio that is fed in would be compared with these codebooks to find the best speaker that fits the comparison with minimal error. This error value is then compared against a predetermined threshold to check if the audio really belongs to the predicted speaker or to an impostor. The entire model was created, trained, and tested with MATLAB.

## 2. FEATURE EXTRACTION

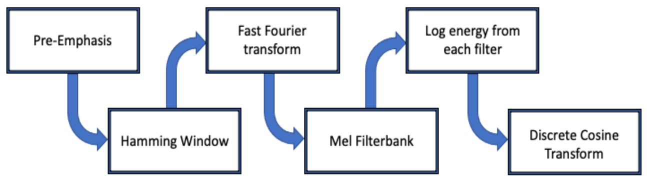


Fig. 2. Feature extraction with MFCCs

Before training any speaker recognition model, a set of features need to be extracted from the audio input. These features serve as parametric representations of acoustic signals that allow for a better recognition performance [5]. One set of such features are the Mel-frequency cepstral coefficients (MFCCs), which are based on human hearing perceptions. The first step taken to obtain MFCCs is to apply a pre-emphasis filter to the audio input.

Pre-emphasis filtering helps compensate for the high-frequency parts of speech that are suppressed when humans produce sound. This also amplifies high-frequency formants that are typically more important for speaker recognition [6,7]. For an audio signal x(n), the pre-emphasized signal y(n) is defined as:

$$y(n) = x(n) - a * x(n - 1) \quad (1)$$

where a is typically a number from 0.9 to 1. A value of 0.98 was chosen for this particular model.

After pre-emphasis, the speech signals are segmented into a series of overlapping frames for processing. A frame size of 20 ms and an overlap of 10 ms was selected for the audio files that were sampled at a frequency of 8000 Hz. The frames are then multiplied with a hamming window to prepare the signal to be converted from the time domain into the frequency domain. The equation for applying a Hamming window is:

$$y(n) = x(n) * w(n) \quad (2)$$

where y(n) represents the signal after the window has been applied, x(n) represents the signal before the window has been applied, and w(n) represents the Hamming window defined as:

$$w(n) = 0.54 - 0.46 * \cos\left(\frac{2\pi n}{N - 1}\right) \quad (3)$$

Here, N represents the number of samples in each frame, which can easily be determined by multiplying the frame size and the sampling frequency.

The fast Fourier transform (FFT) is applied to the windowed signal to obtain the frequency-domain representation of the signal. A bank of triangular Mel-scale filters is applied to this frequency domain representation. The first ten filters are linearly spaced from 0 to 1000 Hz, and the next ten are logarithmically spaced from 1000 Hz to 4000 Hz, as shown in Fig. 3. The discrete cosine transform (DCT) of the logarithm of this Mel-filtered signal is taken to obtain 12 MFCCs. The zeroth term obtained from the DCT is replaced with the log energy value obtained by averaging over the frequency-domain signal representation for the corresponding frame and taking the logarithm of that average value. Once these 13 values (12 MFCCs and one log energy value) have been computed for each frame, the delta MFCCs and the delta log energy are computed by obtaining a locally smooth estimate of the first derivative of the collection of these MFCC and log energy values throughout the frames. This yields an additional 13 values per frame representing all delta values. The double-delta values representing the smooth estimate of the second derivative is computed similarly to yield another set of 12 double-delta MFCCs and one double-delta log energy for each frame. Finally, the MFCC vector comprises of a total of 39 values for each frame as a result. This representation of all the features for a frame of the signal is known as an acoustic vector.

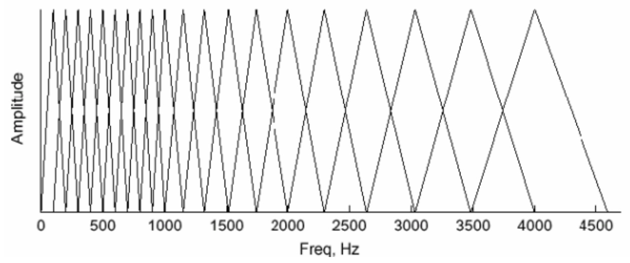


Fig. 3. The Mel-filterbank applied in the frequency domain

## 3. MODEL TRAINING

Many methods exist for training a set of acoustic vectors to recognize a speaker, including relatively more complex methods like Gaussian mixture models, hidden Markov models, and support vector machines [3]. A relatively simpler high-accuracy method that is commonly used for text-dependent speaker recognition with a small number of speakers each with a short password phrase is called vector quantization. Vector quantization is the process of mapping vectors from a large vector space to a finite number of regions (or clusters) in that space [5]. All these clusters are accumulated based on certain vectors (or centroids) that represent them. When stabilized, these vectors are called codewords. The collection of all the codeword vectors is stored in a matrix called the codebook that is unique to each speaker-phrase combination. While many algorithms can be used to yield codebooks, one that is commonly used is called the LBG algorithm, which finds clusters around centroids using the Euclidean distance.

### 3.1. Euclidean distance

The Euclidean distance for two matrices is needed to find which cluster each acoustic vector belongs to. The Euclidean distance is also needed during test time to compare the codebook with the acoustic vectors extracted from an audio input. For the second case, since each test audio file would not yield the exact same number of acoustic vectors every time (due to the duration of the audio being slightly longer or shorter), the Euclidean distance for two matrices of different column dimensions would need to be computed. For two matrices A and B, where A is $MxN_1$, B is $MxN_2$, and $N_1 < N_2$, the Euclidean distance is given by a matrix D of dimensions $N_1$ x $N_2$ defined as:

$$d_{i,j} = \sqrt{\sum_{k=1}^{M} (a_{k,i} - b_{k,j})^2} \qquad (4)$$

where $d_{i,j}$ corresponds to row i and column j of matrix D, $a_{k,i}$ corresponds to row k and column i of matrix A, and $b_{k,j}$ corresponds to row k and column j of matrix B.

## 3.2. The LBG algorithm

The Linde-Buzo-Gray (LBG) algorithm iteratively finds the best centroids for each set of clusters using the Euclidean distance [8]. The process starts with finding a centroid for all of the acoustic vectors for an audio signal. This is typically the mean or the median vector of all the vectors. This represents a one-codeword codebook. Then, based on a splitting parameter ε (chosen to be 0.01 in the model), a new codebook $y_{new}$ generated with twice the number of centroids is defined as:

$$y_{new} = [y * (1 + \varepsilon), y * (1 - \varepsilon)] \qquad (5)$$

where y represents the old codebook (which would have been $y_{new}$ in the previous iteration).

After the codebook is updated, the Euclidean distance between each centroid and all the acoustic vectors is calculated. For each acoustic vector, the centroid that is closest to the vector is assigned to the vector. After this nearest neighbor searching process, the acoustic vectors are separated into n clusters, where n represents the current number of centroids. The smallest distance between a centroid and its closest acoustic vector is stored as the distortion value of that centroid. Now, the mean vectors of each of these new clusters is calculated, and the new centroids get updated to these mean values. This process of finding the nearest neighbors for each acoustic vector and then updating the centroids is repeated until the centroids converge to a fixed vector. This stabilization is determined by the percent change of the distortion values, and the stabilization threshold is typically defined by ε as well.

Once the centroids converge, the next iteration of the algorithm takes place, where all the steps starting from updating the codebook based on (5) are repeated. The iteration stops after the centroids have doubled and stabilized enough times to reach a fixed number of centroids (16 is chosen for this model, but any multiple of two can be chosen depending on the number of acoustic vectors). Fig. 4. shows the updated centroids for four iterations plotted with the 3rd and the 6th MFCCs. Since all 39 dimensions of the acoustic vectors and the codebooks could not be shown, these two coefficients were chosen at random for illustrative purposes.

For each digit from "0" to "9" uttered by each of the 6 speakers in the FSDD, one audio file is taken to train the model by first extracting the acoustic vectors with MFCCs and then generating a speaker-specific and digit-specific

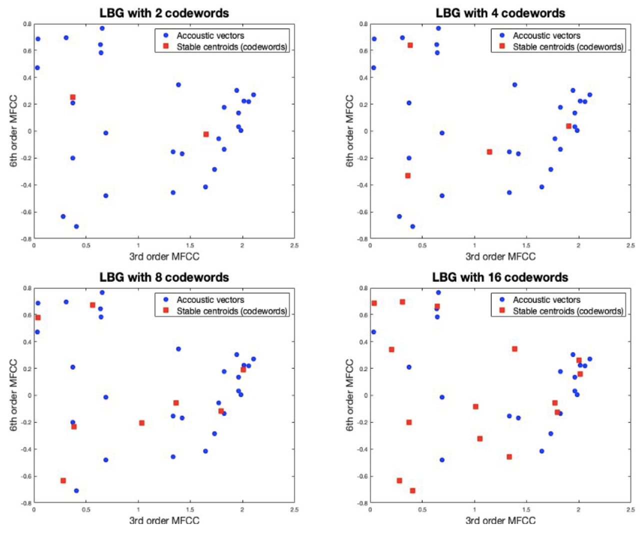


Fig. 4. Multiple iterations of the LBG algorithm

codebook. Since all the speakers say all the digits here, a total of 60 unique codebooks are generated.

## 4. MODEL TESTING

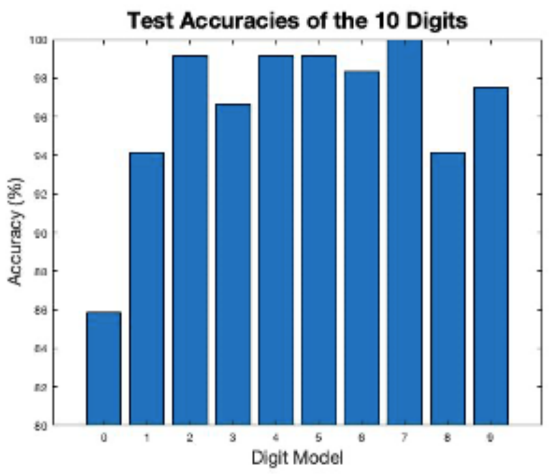
### 4.1. Closed-set speaker recognition

The focus of testing is to observe the ability of the model to discriminate between different speakers even with the same password for all of them. For each digit, the Euclidean distance between each speaker's codebook and the matrix of acoustic vectors is computed and the distortion is found. The minimum distortion value from all the speakers' distortion values is found, and the speaker corresponding to the minimum distortion value is identified as the speaker who spoke the test audio. This particular test assumes a closed-set system, or that all the incoming test audio belongs to one of the 6 speakers and not to any impostors.

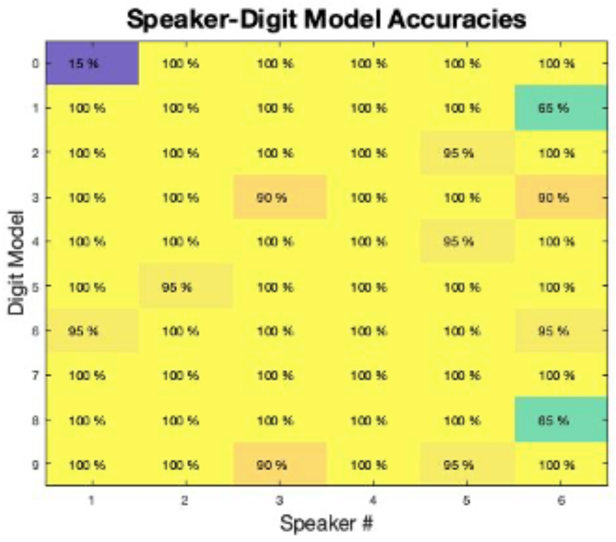
Fig. 5 shows the results from testing 20 samples from all speakers for all digits, for a total of 1200 samples. Based on Fig. 5 (a), the model accuracy seems to be the worst (86 %) for "0" and best for "7" (100 %). A better understanding of this performance can be obtained by observing Fig. 5 (b). The training audio of speaker 1 for "0" seems to perform very poorly (15 % test accuracy), and the most likely cause is that the training "0" is mispronounced by speaker 1 compared to their test "0" audio set. When deployed, the model could keep a running track of performance accuracies and alert the users of a bad training read (to give them an opportunity to re-record the training audio sample) for consistently poor accuracies to fix this type of training mispronounce issue.

### 4.2. Open-set speaker recognition

The open-set model builds on the closed-set model by simply setting a threshold for the minimum distortion value



(a)



(b)

Fig. 5. (a) Bar graph showing the test accuracies for each digit from “0” to “9” (b) Matrix showing a breakdown of speaker-digit test accuracies

calculated during closed-set testing. The speaker corresponding to the minimum distortion value is identified as the test audio speaker if this minimum distortion is less than a predetermined distortion threshold. If this minimum distortion is greater than the threshold, then the test audio speaker is labeled as an impostor.

Fig. 6 shows a DET curve constructed by sweeping the distortion threshold from 2 to 3 with a small step size. For each digit, the model was tested with 6 original speaker audio sets and 6 different impostor sets, with a set having 10 audio files, giving a total of 1200 tested files. The impostor audio was used from the Audio MNIST database [9]. The equal error rate (EER) is 2.2 %, which indicates that it is acceptable to have this kind of distortion threshold implementation for this particular system to identify impostors.

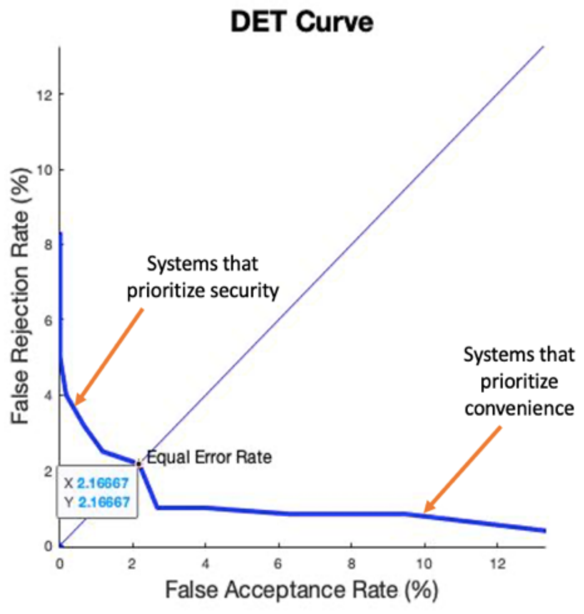


Fig. 6. DET curve with EER and modes of operation

## 5. CONCLUSIONS

The paper provides a model for text-dependent speaker recognition for a small number of speakers with a few seconds of test audio. From a training audio set, MFCCs are extracted for each frame of audio to construct a set of acoustic vectors for each audio file. Using the LBG algorithm, speaker-and-phrase-specific codebooks are generated during the training phase. These codebooks are used to classify unknown test audio speakers (based on the smallest distortion values from the test audio MFCCs to the codebooks) or label them as impostors depending on whether the values are greater than a distortion threshold.

Many existing works only focus on closed-set speaker recognition with vector quantization [2,5,6]. However, the proposed model accounts for open-set speaker recognition by setting a distortion threshold and testing the system. The results are promising, with the EER being less than 5 %. So, the proposed speaker recognition model can be used in place of (or in addition to) a user-ID and password authentication for household applications.

## 6. REFERENCES

[1] R. Rashid, N. Mahalin, M. Sarijari, and A.A. Abdul Aziz, "Security System Using Biometric Technology: Design and Implementation of Voice," Proceedings of the International Conference on Computer and Communication Engineering 2008, Kuala Lumpur, Malaysia, 2008.

[2] H.M.O. Al Marzuqi, S.M. Hussain, and A. Frank, "Device Activation based on Voice Recognition using Mel Frequency Cepstral Coefficients (MFCC's) Algorithm," International Research Journal of Engineering and Technology (IRJET), Vol 06, Issue 03, Muscat, Oman, 2019.

[3] S. Chen and Y. Luo, "Speaker Verification using MFCC and Support Vector Machine," Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS) 2009 Vol I, Hong Kong, 2009.

[4] https://github.com/Jakobovski/free-spoken-digit-dataset. (Free Spoken Digit Database)

[5] A.S. Thakur and N. Sahayam, "Speech Recognition Using Euclidean Distance," International Journal of Emerging Technology and Advanced Engineering (IJETAE), Vol 3, Issue 3, 2013.

[6] K. Chakraborty, A. Talele, and S. Upadhya, "Voice Recognition Using MFCC Algorithm," International Journal of Innovative Research in Advanced Engineering (IJIRAE), Vol 1, Issue 10, Vashi, India, 2014.

[7] L. R.Rabiner and R. W.Schafer, "Digital Processing of Speech Signals," Prentice-Hall, New Jersey, The United States of America, 1978.

[8] Y. Linde, A. Buzo, R. Gray, "An Algorithm for Vector Quantizer Design," IEEE Transactions on Communications, Vol 28, Issue 1, 1980.

[9] S. Becker, M. Ackermann, S. Lapuschkin, K. Müller, and W. Samek, "Interpreting and Explaining Deep Neural Networks for Classification of Audio Signals," CoRR, Saarbrücken, Germany, 2018.