A GAUSSIAN MIXTURE MODEL BASED SPEECH RECOGNITION SYSTEM USING MATLAB

Manan Vyas

B.E Electronics, University of Mumbai
mananvvyas@gmail.com

ABSTRACT

This paper aims at development and performance analysis of a speaker dependent speech recognition system using MATLAB®. The issues that were considered are 1) Can Matlab, be effectively used to complete the aforementioned task, 2) Accuracy of the Gaussian Mixture Model used for parametric modelling, 3) Performance analysis of the system, 4) Performance of the Gaussian Mixture Model as a parametric modelling technique as compared to other modelling technique and 5) Can a Matlab® based Speech recognition system be ported to a real world environment for recording and performing complex voice commands. The aforementioned system is designed to recognize isolated utterances of digits 0-9. The system is developed such that it can easily be extended to multisyllabic words as well.

KEYWORDS

Automatic Speech Recognition (ASR), Feature Extraction, Fast Fourier transform, Discrete Cosine Transform, Linear Prediction (LPC), Mel Frequency Cepstral Co-efficient (MFCC), Gaussian Mixture Model (GMM).

1. INTRODUCTION

Speech recognition is the process of automatically recognizing who is speaking and what is spoken based on unique characteristics contained in speech signal. This technique makes it possible to use the speaker's voice and the spoken word to verify their identity and control access to services such as voice dialling, database access services, information services, voice mail, security control for confidential information areas, and remote access to computers and an aid to the physically challenged.

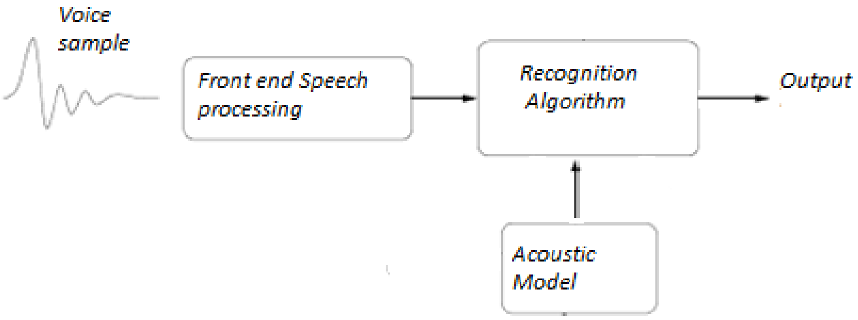


Figure 1. Block Diagram of the Developed Model

DOI : 10.5121/sipij.2013.4409

The developed system at the highest level contains the Feature extraction & training and Feature matching modules. Feature extraction, a component of the front end speech processing block, is the process of extracting unique information from voice data that can later be used to identify the speaker. The Gaussian Mixture Model was chosen as the underlying acoustic model for the system wherein the extracted features were modelled using multi-component Gaussian PDF. The recognition algorithm compares the real-time voice input by the user and implements the actual procedure of identifying the speaker by comparing the extracted data from the real-time voice input with a database of known speakers and their word based dictionary. Therefore, the system needs to be trained beforehand for developing the word-bank as a reference for matching algorithms.To perform this, a series of acoustic features are extracted from the speech signal, and then recognition algorithms are used. Thus, the choice of acoustic features is critical for the system performance [1]. If the feature vectors do not represent the underlying content of the speech, the system will perform poorly regardless of the algorithms applied [2]. And thus the Mel-Frequency Cepstrum Co-Efficient (MFCC) technique for feature extraction was used. Moreover, the production of speech signal is approximated to a source-filter model where the speech source excites the vocal tract filter.

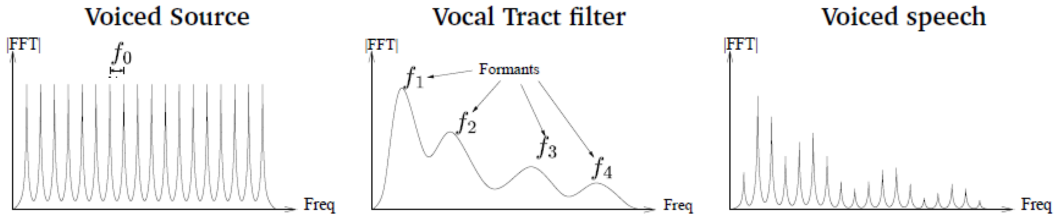


Figure 2. Source-Filter Model of Speech Production.

2. Voice Activity Detection

A utility function was written in Matlab® to detect an input speech signal. It accepts input speech of repeated utterances of a single digit by the user through a microphone and returns a plot indicating detected voice activity. Input speech is sampled at 8 kHz. Frames are then created containing 160 contiguous samples with 80 sample overlap and further processing is carried out taking one frame as a single computational unit. In order to enable voice activity detection in noisy environments, user-defined parameters of std_energy and std_zero_crossings can be set to [0, 1] with zero being a noise free environment and 1 being extremely noisy. For room environment conditions, these parameters were set to 0.5 each as default. A temporary buffer of length equal to 160 samples was created for comparison between successive frames. The utility function carries out the following mentioned steps,

1. Calculate the energy and number of zero crossings in the frame
2. Compare these values with threshold (std_energy & std_zero_crossings) to determine if we have possible voice activity and hence a spoken digit.
3. If either energy or zero crossings exceeds the threshold, continue analysing frames and start buffering.
4. If number of contiguous frames does not exceed "bufferlength", we have a false alarm. Continue analysing frames and go back to 1.
5. If number of contiguous frames exceeds "bufferlength", we have detected a word. Continue buffering and analysing frames.
6. Keep analysing frames until we encounter continuous frames where neither energy nor zero crossing exceeds the threshold or silence is detected. This means we have analysed past the end of the spoken digit.

7. Compare duration of detected digit with time threshold (set to 0.25s). If duration exceeds threshold, mark voice activity. If duration does not exceed threshold, disregard digit. Continue analysing frames and go back to (1). The cycle continues until all frames are analysed.

The utility function produces the following output.

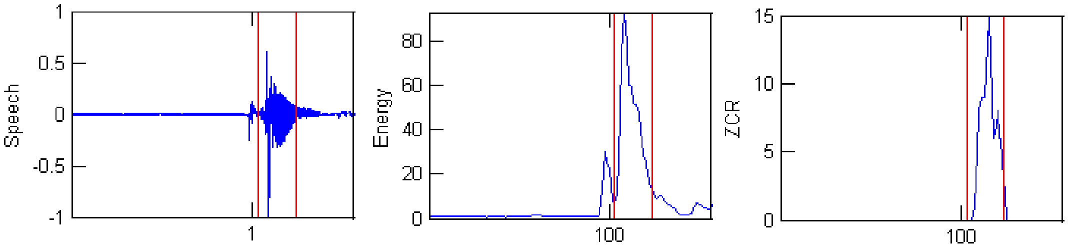


Figure 3. Output produced by the voice detection function of a single utterance of digit „one" by user Manan. The speech signal is represented by the *blue* plot, *red plot* detects voice activity.

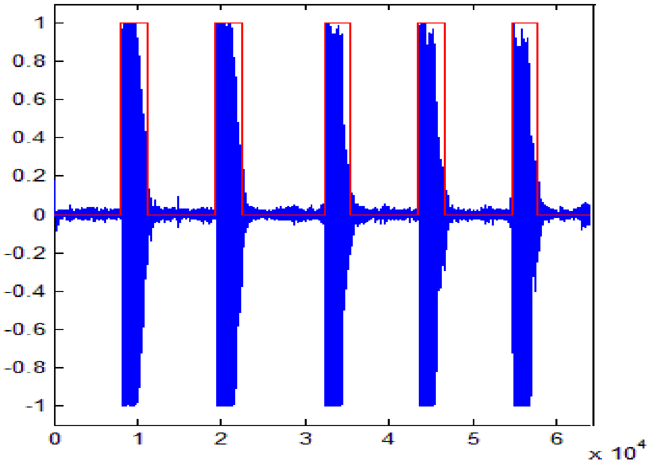


Figure 4. Output produced by the voice detection function of repeated utterance of digit "zero" by user. The blue plot indicating the speech sample and the red plot indication spoken segments in the sample.

3. Feature Extraction

A voice feature is merely a matrix of numbers in which each number represents the energy or average power that is heard in a particular frequency band during a specific interval of a speech signal [3]. It is assumed that the speech is close to stationary during this short period of time because of the relatively limited flexibility of the throat. Before picking out our features from the frequency domain, but before we get there by taking the fast Fourier transform, we multiply by a windowing function to reduce spectral leakage. In order to choose the most optimum windowing function, the following parameters were considered,

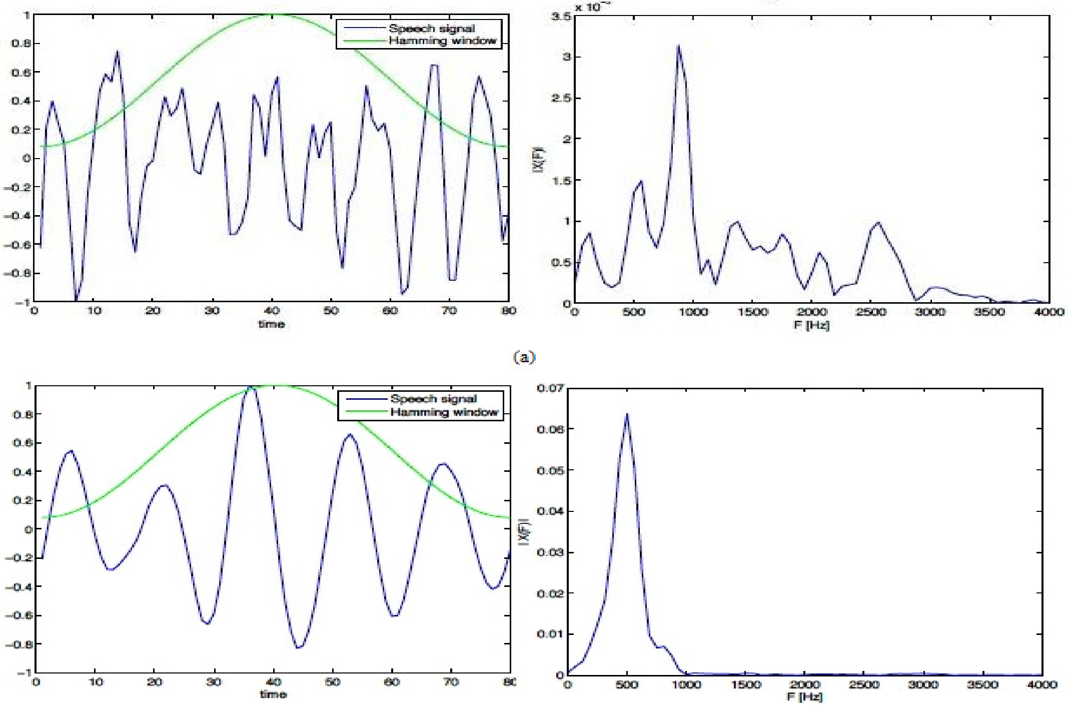
Time to compute: The computation time required by Matlab® to generate feature vectors after voice detection was carried out. This was achieved using the tic ( ) and toc ( ) built-in of Matlab®.

Probability of Error: Gives a numerical estimation of the amount of energy (E) of noise within the speech signal detected by the voice activity detection utility function. It was assumed that at least 50 frames comprised of ambient noise for uniform comparison of different window function. It was calculated using the following equation,

$$P_e = erfc\sqrt{\frac{E}{50}} \qquad \text{........................ (1)}$$

Table 1. Performance analysis of different windowing functions.

Window Function	**Size (N)**	**Time to Compute (sec)**	**Probability of error in output frame**
Rectangular	64	0.772	0.0724
Hamming	64	0.509	0.0297
Hanning	64	0.397	0.0610
Blackmann	64	0.607	0.0483



(a)

(b)

Figure 5 (a): Hamming Window (green plot) excerpt of a 80msec unvoiced segment of speech signal (blue plot LHS) indicating the presence of noise at higher frequencies (blue plot RHS) in the single sided magnitude spectrum of the same speech signal multiplied by Hamming window. 5 (b): Hamming Window (green plot) excerpt of an 80msec voiced segment of speech signal (blue plot LHS) indicating the dominant speech component at lower frequencies (blue plot RHS) in the single sided magnitude spectrum of the same speech signal multiplied by Hamming window.

Thus, it is observed that unvoiced segments of speech signal (containing mostly the ambient noise) produces a spectrum containing peaks at higher frequencies after windowing and the voiced segments majorly produces a single dominant spectrum at lower range of the frequency spectrum after windowing. This is more pronounced in the PSD plot of the unvoiced segment as follows,

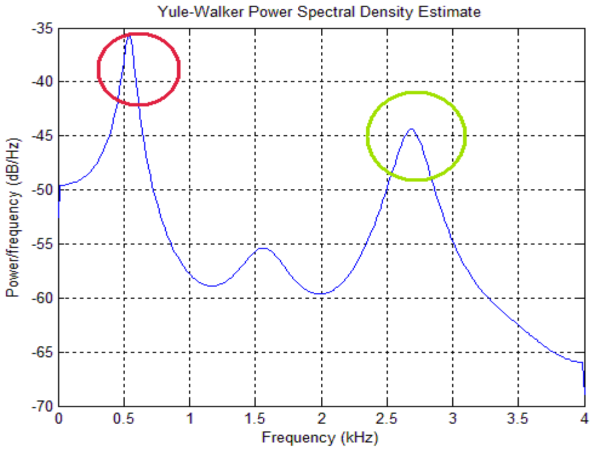


Figure 6. Single sided PSD estimate of the digit “eight“ spoken by user representing dominant frequency components of noise (circled Green) and speech signal (circled Red).

This dominant noise component within the speech signal was now filtered out using an appropriate LPF of upper cut off frequency of 2.2 kHz. Thus a significant chunk of noise was removed from the speech sample before carrying out further processing.

Feature extraction is the process of extracting unique information from voice data that can later be used to identify the speaker. The Mel Frequency Cepstrum Co-Efficients (MFCC) technique was used in this project. The MFCC technique is based on the known variation of the human ear’s critical bandwidth frequencies with filters that are spaced linearly at low frequencies and logarithmically at high frequencies to capture the important characteristics of speech.This is done by multiplying the output spectrum of the filter with the set of triangular weights as shown below,

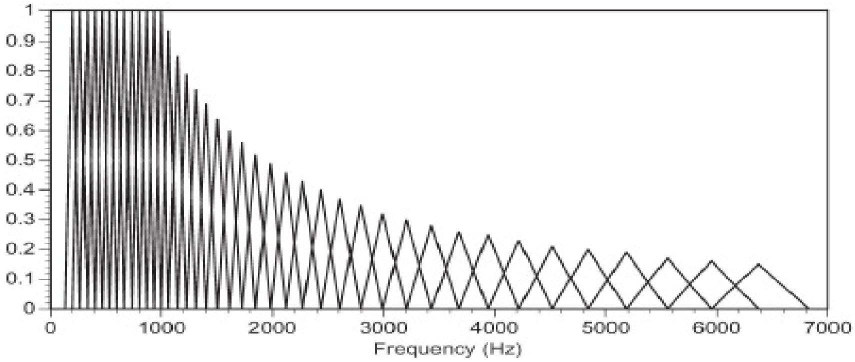


Figure 7. The Mel Scale

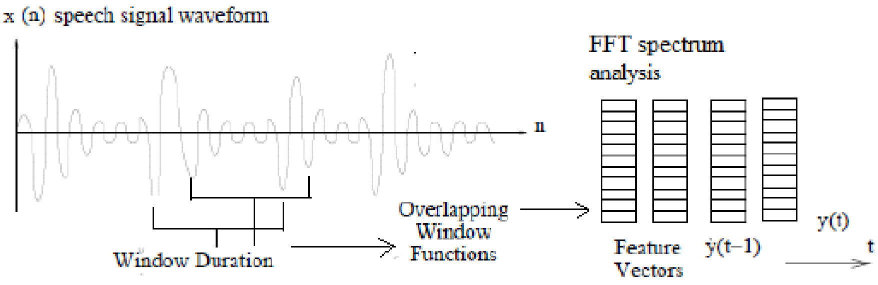


Figure 8. Feature Extraction Procedure

3.1 Calculating Mel Frequency Cepstrum Co-Efficients

Input voice sample in the project is segmented in frames of 30 ms, and the window analysis is shifted by 10 msec. Each frame is converted to 13 MFCCs plus a normalized energy parameter. The first and second derivatives (Δ and Δ2) of MFCCs and energy are estimated, resulting in 39 numbers representing each frame. A sampling rate of 8 kHz is used for the input voice sample and hence, for each 10 ms the feature extraction module delivers 39 numbers to the statistical analysis stage. This operation with overlap among frames is equivalent to taking 80 speech samples without overlap and representing them by 39 numbers. In fact, with each speech sample represented by one byte and each feature is represented by four bytes, the parametric representation increases the number of bytes to represent 80 bytes of speech to 136 bytes. The resultant matrices are referred to as the Mel-Frequency Cepstrum Coefficients. This spectrum provides a fairly simple but unique representation of the spectral properties of the voice signal which is the key for representing and recognizing the voice characteristics of the speaker. After the speech data has been transformed into MFCC matrices, one must apply one of the widely used pattern recognition techniques to build speaker recognition. models using the data attained in the feature extraction phase and then subsequently identify any sequence uttered by an unknown speaker. The reference voiceprint with the lowest distance measured from the input pattern was deemed the identity of the unknown speaker. The distance metric used within the system was the Euclidean distance measure. The Euclidean distance between two points is the minimum distance between the points in any single dimension. The distance between points X = (X1, X2 … etc.) and Y = (Y1, Y2 … etc.) is computed: $d(X.Y) = MIN\,|Xi - Yi|^2$ .The following steps were followed to extract Mel Frequency Cepstrum Co-Efficients MFCC from the speech signal.

1. Take the Fourier transform of (a Hamming windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the Mel scale, using triangular overlapping windows.
3. Take the logs of the powers at each of the Mel frequencies.
4. Take the discrete cosine transform of the list of Mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

4. PARAMETRIC MODELLING USING GAUSSIAN MIXTURE MODELS (GMM)

A Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. GMMs are commonly used as a parametric model of the probability distribution of continuous measurements or features in a biometric system, such as vocal-tract related spectral features in a speaker recognition system. GMM parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm or Maximum *A Posteriori* (MAP) estimation from a well-trained prior model. The Gaussian mixture model for speech representation assumes that a M component mixture model with windowing function weights P(ωm) and the mixture components in the input voice sample contains Gaussian components given by,

$$p(x|\omega_m) = \frac{1}{\sqrt{2\pi\sigma_m^2}} e^{[-\frac{(x-\mu_m)^2}{2\sigma_m^2}]}$$

Where is the μm is the mean and σm is the standard deviation for component p(ωm).

The complete Gaussian mixture model is parameterized by the mean vectors, covariance matrices and mixture weights from all component densities. GMMs are often used in biometric systems, most notably in speaker recognition systems, due to their capability of representing a large class of sample distributions. Given training vectors consisting of MFCC feature vectors and a GMM configuration, we wish to estimate the parameters of the GMM, λ, which in some sense best matches the distribution of the training feature vectors. There are several techniques available for estimating the parameters of a GMM [4]. By far the most popular and well-established method is maximum likelihood (ML) estimation. The aim of ML estimation is to find the model parameters which maximize the likelihood of the GMM given the training data.

For a sequence of T training vectors X = {x1, . . . , xT }, the GMM likelihood, assuming independence between the input data vectors, can be written as,

$$P(X|t) = \prod_{t=1}^{T} P(x_t|\lambda)$$

Unfortunately, this expression is a non-linear function of the parameters λ and direct maximization is not possible and hence the iterative Expectation-Minimization algorithm (EM) was used that was then used for training as well as matching purposes.

The E-step comprised of,

(1)The total likelihood with g being the Gaussian PDF

$$t_j = \sum_{i=1}^{N} pi \;\; g(x_j, \mu_i, \Sigma_i) \; j = 1 \; to \; N$$

(2)The normalized likelihoods

$$n_{ij} = p_i \; \frac{g(x_j, \mu_i, \Sigma_i)}{t_j} \; ; i = 1 \; to \; j-1$$

(3)The notional counts $C_i = \sum_{j=1}^{N} n_{ij}$

(4)The notional means $\widehat{x_i} = \sum_{j=1}^{N} (n_{ij} \; x_i) \frac{1}{C_i}$

(5)The notional sums of squares $S\,S_i^{pq} = \sum_{j=1}^{N} (x_i^p \, x_j^q \; n_{ij}) \; \frac{1}{C_i}$; p ,q =1,2,3.. for multi-component Gaussian

The M-step is an updating step wherein,

(1) $p_i = C_i / S_i$

(2) $\mu_i = \widehat{x_i}$

(3) $\Sigma_i^{pq} = SS_i^{pq} - x_i^p \, x_i^q \; ; i = 1 \; to \; N$

The extracted MFCC features were then fitted into a 3-component Gaussian PDF as follows

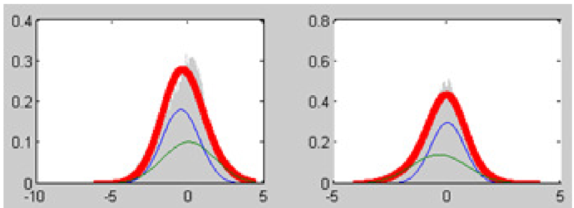


Figure 9. Fitting the MFCC elements of uttered digit ‘five’ into Gaussian distribution curve for 2 different users.

Thus, the following underlying probabilistic measure became the underlying model for the input utterances during training which then can be stored in the system as a reference for matching when the system was ported into real-time environment,
Mixture Weights:

$$\bar{w}i = \frac{1}{T}\sum_{t=1}^{T} P(i|x_t)$$

Mean:

$$\mu i = \frac{\sum_{t=1}^{T} P(i|x_t,\lambda\,)x_t}{\sum_{t=1}^{T} P(i|x_t,\lambda\,)}$$

Diagonal Co-Variance:

$$\sigma i^2 = \frac{\sum_{t=1}^{T} P(i|x_t,\lambda\,)x_t^2}{\sum_{t=1}^{T} P(i|x_\iota,\lambda\,)} - \mu i$$

And, *a posteriori* probability for component i is given by:

$$P(x_t|\lambda) = \frac{\mathrm{wi\ p}(x_t|\mu i,\sigma i^2)}{\sum_{\mathrm{k}=1}^{\mathrm{M}} \mathrm{wk\,p}(x_t|\mu i,\sigma i^2)}$$

The same iterative steps were carried out for matching purposes in real-time and the Euclidean distance between the word-bank contents and real-time input was calculated and a correct match is said to be found when, Match in word bank = Min | [number of hidden states, frequencies extracted (real time input)], [number of hidden states, frequencies extracted (training set in word bank)] | was obtained.

5. RESULTS AND DISCUSSION

- A basic speech recognition system which recognises a digit from 0 to 9 was thus developed. The system was trained in an environment with minimum ambient noise. One of the shortcomings of the system is that the performance degrades in the presence of noise. The system gives the most accurate results when implemented in the environment where it was trained.
- The system performance was analysed by increasing the number of training iterations for the EM algorithm, including setting a threshold on the likelihood difference between steps. That, however, proved to have little benefit in practice; neither the execution time nor the amount of misclassification rate showed any mentionable improvements over just fixing the number of iterations. The reason why the execution time did not show any significant improvements is because most of the execution time is spent during feature extraction, and not in training.
- It is also important to note that when the number of Gaussian components are < 2, there are many `six's' misclassified as unvoiced segments of speech sample, and vice versa, due to the loss of temporal information.
- Another important parameter is the number of samples in each frame. If the frame is too small, it becomes hard to pick out meaningful features, and if it is too large, temporal information is lost. Hence an optimal 160 contiguous frames with 80 sample overlap was chosen, which is the case with most speech recognition systems.
- The concatenation of the training examples trains *a posteriori* probability involving silent segments between two consecutive utterances that is not needed for classification. The EM

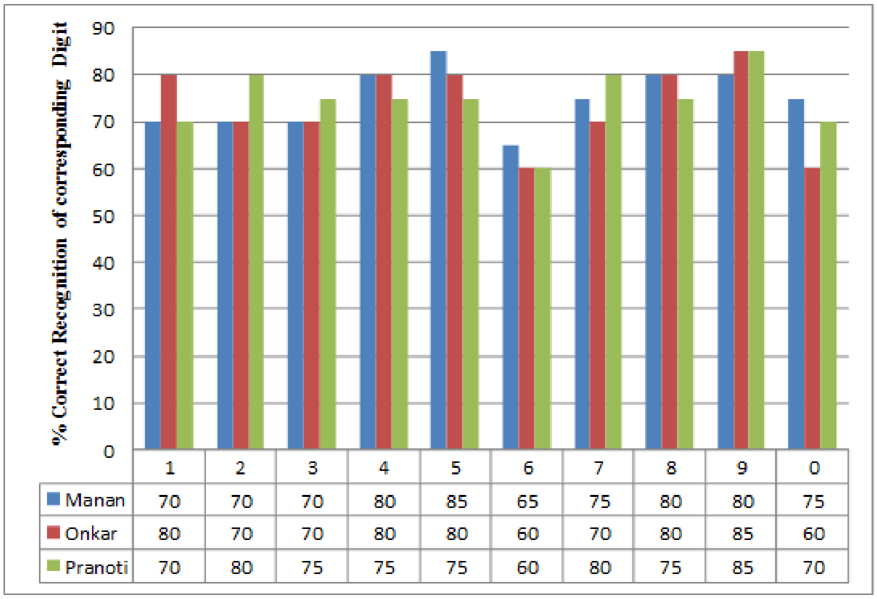
algorithm can be modified for multiple training examples such that concatenation is not necessary, but that was not implemented during this project as the concatenation of contiguous frames while feature extraction worked well.

- The system is platform/hardware independent. We have developed the system in MATLAB® which can be implemented on any laptop/home desktop irrespective of the configuration. But the system needs to be trained on the same in the environment where it needs to be implemented for it to work.
- The system was also trained to recognise words for e.g. hello etc. It was observed that as the complexity of the spoken word increases; complexity implies more than one dominant syllable in the word i.e. multi syllabic words; the data generated for comparison of real time input and the stored word increases multifold. The parameters extracted from EM algorithm as well the means and diagonal co-variances for such words become too complex, inaccurate with a large processing time for analysis. The performance of system degrades.
- The statistical parameters obtained after training the system for digit 5 & 6 are too close. Therefore if the digit is not spoken clearly during recognition, the system falters. The digit 6 gives the lowest accuracy, the reason being the speech sample for 6 has the highest amount of “unvoiced” speech signal. Therefore it is treated as unvoiced speech data i.e. a hiss than voiced data.
- The system supports multiple users. Rigorous training has to be carried out for each user. The system is then able to identify the voice sample and the spoken digit of a particular user from the users. This is a significant deviation in the positive direction from the current trend to develop speaker independent recognition system.

6. CONCLUSION

It is concluded that the use of GMM as a statistical model in the speech recognition gives a very high accuracy of recognition (> 70%).

Table 2. Qualitative analysis of GMM documented by mapping recognition accuracy of each digit for the said user wherein the system was trained in laboratory environment and tested by the same user in the same environment.



	1	2	3	4	5	6	7	8	9	0
🟦 Manan	70	70	70	80	85	65	75	80	80	75
🟥 Onkar	80	70	70	80	80	60	70	80	85	60
🟩 Pranoti	70	80	75	75	75	60	80	75	85	70

6.1 Future scope and applications:

- Hardware implementation of the system using a Digital Signal Controller (DSC).
- The next step would be to recognise continuous speech rather than isolated digit recognition.

The following are some suggested applications for the developed system,

- Security Systems.
- Healthcare: Electronic Medical Records
- Home Automation
- Telephony
- Robotics
- Hands Free Computing and aid to the physically handicapped.

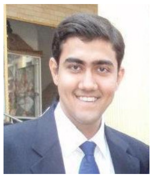
REFERENCES

[1] X.Huang, A. Acero, and H.-W. Hon, “Spoken Language Processing: A Guide to Theory, Algorithm and System Development”. Prentice Hall PTR May 2001

[2] Matthew Nicholas Stuttle, “A Gaussian Mixture Model Spectral Representation for Speech Recognition”. Hughes Hall and Cambridge University Engineering Department. July 2003

[3] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, pp. 257-286, Feb 1989.

Author



Manan Vyas received his Bachelor of Engineering in Electronics degree from University of Mumbai in July 2012. He has also completed MITx 6.002 – a pilot course on Circuits and Electronics by Massachusetts Institute of Technology with an A grade. He is also a recipient of the J.R.D Tata Scholarship for excellent academics during his engineering. His passions include playing football and trekking.