Design of Keyword Spotting System

Based on Segmental Time Warping of Quantized Features

**A Thesis**

**Submitted to**

**the Temple University Graduate Board**

**In Partial Fulfillment**

**Of the requirements for the Degree**

**Masters of Science in Electrical Engineering**

**By**

**Piush Karmacharya**

**January 2012**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Dr. Robert Yantorno**

**Thesis Advisor**

**Dr. Joseph Picone Dr. Dennis Silage**

**Committee Member Committee Member**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Abstract

Keyword Spotting in general means identifying a keyword in a verbal or written document. In this research a novel approach in designing a simple spoken Keyword Spotting/Recognition system based on Template Matching is proposed, which is different from the Hidden Markov Model based systems that are most widely used today. The system can be used equally efficiently on any language as it does not rely on an underlying language model or grammatical constraints.

The proposed method for keyword spotting is based on a modified version of classical Dynamic Time Warping which has been a primary method for measuring the similarity between two sequences varying in time. For processing, a speech signal is divided into small stationary frames. Each frame is represented in terms of a quantized feature vector. Both the keyword and the speech utterance are represented in terms of 1-dimensional codebook indices. The utterance is divided into segments and the warped distance is computed for each segment and compared against the test keyword. A distortion score for each segment is computed as likelihood measure of the keyword. The proposed algorithm is designed to take advantage of multiple instances of test keyword (if available) by merging the score for all keywords used.

The training method for the proposed system is completely unsupervised, i.e., it requires neither a language model nor phoneme model for keyword spotting. Prior unsupervised training algorithms were based on computing Gaussian Posteriorgrams making the training process complex but the proposed algorithm requires minimal training data and the system can also be trained to perform on a different environment (language, noise level, recording medium etc.) by re-training the original cluster on additional data.

Techniques for designing a model keyword from multiple instances of the test keyword are discussed. System performance over variations of different parameters like number of clusters, number of instance of keyword available, etc were studied in order to optimize the speed and accuracy of the system. The system performance was evaluated for fourteen different keywords from the Call-Home and the Switchboard speech corpus. Results varied for different keywords and a maximum accuracy of 90% was obtained which is comparable to other methods using the same time warping algorithms on Gaussian Posteriorgrams. Results are compared for different parameters variation with suggestion of possible improvements.

Table of Contents

[Abstract 1](#_Toc300569591)

[List of Figures 4](#_Toc300569592)

[List of Tables 6](#_Toc300569593)

[1 Introduction to Keyword Spotting 7](#_Toc300569594)

[2 Research Objective 10](#_Toc300569595)

[3 Background Study 12](#_Toc300569596)

[3.1 Speech Features 13](#_Toc300569597)

[3.1.1 Linear Prediction Coding (LPC) 13](#_Toc300569598)

[3.1.2 Mel-Frequency Cepstrum Coefficients (MFCC) 15](#_Toc300569599)

[3.1.3 Other Features 18](#_Toc300569600)

[3.2 Common Approaches to Keyword Spotting 18](#_Toc300569601)

[3.2.1 Dynamic Time Warping (DTW) 19](#_Toc300569602)

[3.2.2 Hidden Markov Models 21](#_Toc300569603)

[3.2.3 Other Approaches 23](#_Toc300569604)

[3.3 Previous Research based on Segmental-DTW 24](#_Toc300569605)

[4 Initial Research 29](#_Toc300569606)

[4.1 Cross-Correlation 29](#_Toc300569607)

[4.2 Short-Time Fourier Transform (STFT) Analysis 32](#_Toc300569608)

[4.3 Dynamic Time Warping on LPCC features 33](#_Toc300569609)

[4.4 Keyword Statistics 38](#_Toc300569610)

[5 Keyword Spotting System Design 40](#_Toc300569615)

[5.1 System Overview 40](#_Toc300569616)

[5.2 System Training 41](#_Toc300569617)

[5.2.1 Cluster Training 41](#_Toc300569620)

[5.2.2 Codebook Generation 44](#_Toc300569621)

[5.2.3 Distance Matrix Computation 44](#_Toc300569622)

[5.3 Keyword Detection 46](#_Toc300569623)

[5.3.1 Segmental – Dynamic Time Warping 47](#_Toc300569624)

[5.3.2 Decision Logic 48](#_Toc300569625)

[5.4 Experimental Setup 49](#_Toc300569626)

[5.4.1 Corpus 49](#_Toc300569627)

[5.4.2 Pre-Emphasis 50](#_Toc300569628)

[5.4.3 Feature Extraction 50](#_Toc300569629)

[5.4.4 Quantization 50](#_Toc300569630)

[5.4.5 Keyword Template generation 52](#_Toc300569631)

[5.4.6 Utterance Processing 52](#_Toc300569632)

[5.4.7 Likelihood Comparison 52](#_Toc300569633)

[6 Results 56](#_Toc300569634)

[6.1 Decision Criteria 60](#_Toc300569635)

[6.2 System Operating Characteristic 60](#_Toc300569636)

[6.2.1 Detection Count 60](#_Toc300569637)

[6.2.2 Segment Span 60](#_Toc300569638)

[6.3 System Evaluation 64](#_Toc300569639)

[7 Conclusion 68](#_Toc300569640)

[8 Future Work 70](#_Toc300569641)

[References 71](#_Toc300569642)

# List of Figures

[Figure 1.1. Confusion matrix representing all possible outcome of a binary classifier. 8](#_Toc300569643)

[Figure 1.2. ROC space. 8](#_Toc300569644)

[Figure 3.1. Speech generation approximation based on LPC. 13](#_Toc300569645)

[Figure 3.2. Equal Loudness Curves (also called the Fletcher–Munson curves). 14](#_Toc300569646)

[Figure 3.3. Normalized Mel and Bark scaled frequency. 15](#_Toc300569647)

[Figure 3.4. Speech signal (above) and framwise FFT Magnitude (below). 16](#_Toc300569648)

[Figure 3.5. Different implementations of Mel filter banks. 17](#_Toc300569649)

[Figure 3.6. Dynamic Time Warping. 19](#_Toc300569650)

[Figure 3.7. Two state Hidden Markov Model with hidden states q1, q2 and observation x. 21](#_Toc300569651)

[Figure 3.8. Distribution of the first dimension of MFCC feature vectors for digit ‘ONE’. 25](#_Toc300569652)

[Figure 3.9. Approximation of feature vectors using weighted sum of Gaussians. The solid green line shows the best estimate. 25](#_Toc300569653)

[Figure 3.10. A grayscale representation of phonetic posteriorgram representation for the spoken phrase ‘basketball and baseball’. 26](#_Toc300569654)

[Figure 4.1. Cross-correlation (bottom) of utterance ‘I JUST WANT TO SAY THANK YOU’ (top) and ‘THANK YOU’ (middle) portion of the same (sample 9500-15000 of the utterance marked by the circle). 29](#_Toc300569655)

[Figure 4.2. Cross-correlation (bottom) of same utterance (top) and different instance of phrase ‘THANK YOU’ (middle). The circle highlights the keyword portion in the utterance. 30](#_Toc300569656)

[Figure 4.3. Correlation between the utterance ‘THEY JUST RECORDING OUR CONVERSATION THEY USE IT LIKE’ and keyword ‘CONVERSATION’ spoken by same person. The circle highlights keyword poriton in the utterance. 31](#_Toc300569657)

[Fiugre 4.5.a. STFT analysis for a portion of two instance of keyword REALLY. 32](#_Toc300569658)

[Figure 4.5.b. STFT comparison for portions of two instances of the words REALLY. 33](#_Toc300569659)

[Figure 4.6. Distance between various keywords computed along warped path. 34](#_Toc300569660)

[Figure 4.8. Keyword detection using a threshold value. Figure 4.8.a shows segment of the utterance with keyword CONVERSATION highlighted on red and Figure 4.8.b. shows the segmental distance with threshold indicated by the green horizontal line. 36](#_Toc300569661)

[Figure 4.9. Euclidean Distance plot for entire utterance and the keyword CONVERSATION in grayscale. 37](#_Toc300569662)

[Figure 4.10. Utterance segment that sounds like ‘KINGWISHA’. 39](#_Toc300569663)

[Figure 4.11. Utterance ‘You have to speak English for some’. The keyword ‘ENGLISH’ is distinct here when heard with the context. 39](#_Toc300569664)

[Figure 5.1. Phoneme Cluster. Feature space divided into three phonemes /ER/, /SH/, /AA/. There is no clear separation at the boundary condition and a feature belonging to one class might fall into other. 40](#_Toc300569665)

[Figure 5.2. 2-D Vector Quantization – Feature space quantized into clusters (5) each represented by a centroid. The red and green dots represnts centroids after first and second iterations. 43](#_Toc300569666)

[Figure 5.3. Likelihood Estimation by distance measure. 45](#_Toc300569667)

[Figure 5.4. Distance matrix for a codebook with 6 clusters. 45](#_Toc300569668)

[Figure 5.5. sample likelihood matrix for quantized keyword and utterance. 46](#_Toc300569669)

[Figure 5.6. Segmental DTW for adjustment window size 2. 47](#_Toc300569670)

[Figure 5.7. Sample Cluster centroids for cluster size k=10. 51](#_Toc300569671)

[Figure 5.8. Distance matrix for cluster shown in Figure 5.7. 51](#_Toc300569672)

[Figure 5.9. Frame-wise distance between keyword and utterance divided into segments. The dark green blocks represent the center of each segment separated by dark line. 53](#_Toc300569673)

[Figure 5.10. Minimal distance path for each segment shown by red line. Total distance is the sum of distance along these path divided by keyword length. 53](#_Toc300569674)

[Figure 5.11. Illustration of Keyword detection by voting based approach. If a segment is detected for more than half the keyword templates (n ≥ N/2, N = 6), it is considered a keyword. 54](#_Toc300569675)

[Figure 6.1. Distortion plot for each keyword template (1-8) and the average distortion score from all keyword templates (9). The lowset discortion segment indicated by the green arrow (6.1.9) is the lcoation of the keyword 57](#_Toc300569676)

[Figure 6.2. Distortion score plot for each keyword (a-d) and the average for utterance with no keyword (e). 58](#_Toc300569677)

[Figure 6.3. Average Distortion score plot utterance with two keywords. 59](#_Toc300569678)

[Figure 6.4. Distortion score for utterance with keyword. 59](#_Toc300569679)

[Figure 6.5. Utterance segmentation for R=3. Segment width Ws=2\*R+1=7. 61](#_Toc300569680)

[Figure 6.6. System Performance as a function of segment span – R. (a.) Hits vs R, (b) Misses vs R, and (c) False Alarms vs R. 63](#_Toc300569681)

[Figure 6.7. Execution time per keyword template per minute of utterance. 64](#_Toc300569682)

# List of Tables

[Table 4.1. Different threshold for different keywords. 34](#_Toc300569683)

[Table 4.2. List of keywords the system was tested on. 38](#_Toc300569684)

[Table 6.1. Detection result for cluster of size 512 and segment span of 5. 65](#_Toc300569685)

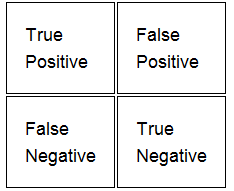
[Table 6.2. Results for different keywords sorted in terms of accuracy. 66](#_Toc300569686)

# Introduction to Keyword Spotting

Keyword spotting (KWS) is a branch of speech processing that deals with identifying certain keywords in a long utterance. Today, people are working on making human-machine interaction seamless and natural. In this context, challenges in this field are even bigger because human conversation contains not only irrelevant words, but also non-intentional sounds like cough, exclamations and noise. If only the embedded information can be extracted by some means, computation can be much more efficient and robust. However people do not speak in terms of isolated words; there are no distinct word boundaries in speech which makes recognition more difficult. Moreover there is always some variation in a keyword every time it is spoken even for the same person. Variations can be in prosody and/or rate of speaking. These are the kinds of challenges a keyword spotting system has to overcome. Some examples of these kinds of systems can be voice dialing, voice command systems, audio search engines, etc. Keyword spotting also has many direct applications such as audio document retrieval and covert speech surveillance systems, etc.

Keyword spotting systems can be divided into two categories - speaker dependent and speaker independent. For speaker dependent systems, models are developed for a specific speaker and hence are not meant to work for other speakers. Speaker independent systems need to be more generic and hence need more complex design. Unlike handwriting verification, speech keyword spotting systems tend to be much more complicated because of the large variation of pronunciations, even from the same speaker, depending on the context and mood of the speaker.

A keyword spotting system can be considered as a binary classifier in which the outcomes are either positive (p) or negative (n). Therefore there are four possible outcomes – a keyword is detected where it is actually present, called a true positive (TP) or a hit; a keyword is detected where it is not present, called a false positive (FP) or a false alarm; a keyword is not detected where it is not present called a true negative (TN); and lastly a keyword is not detected even where it is present, called a false negative (FN) or a miss. The four outcomes can be shown in a 2 by 2 confusion matrix as shown below in Figure 1.1.



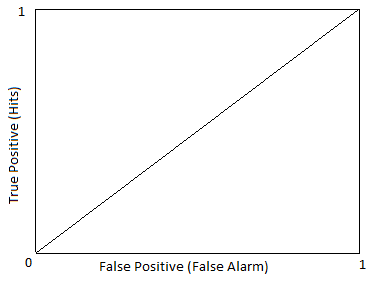
##### Figure 1.1. Confusion matrix representing all possible outcome of a binary classifier.

The accuracy of a binary classifier is defined by the relation:

Accuracy =

= Equation 1.1

The confusion matrix can be used to derive several other evaluation metrics. One of the most commonly used is called the Receiver Operating Characteristic curve also known as the ROC curve. ROC curve considers only the true positive rate (hits) and the false positive rate (false alarms). It is basically a plot of Hits vs. False Alarms as shown below in Figure 1.2.



Worse

Better

##### Figure 1.2. ROC space.

In Figure 1.2 above, he diagonal line divides the ROC space. Operation points above the diagonal line are considered good while operation points below the diagonal are considered poor. Optimal design aims to maximize detection rate while keeping false alarm rate as low as possible. The ideal system would yield a point in the upper left corner, coordinate (0,1) in Figure 1.2, representing 100% accuracy.

There are various approaches to keyword spotting, the most common being Hidden Markov Models (HMMs), Template Matching (TM) and Neural Networks. HMM systems are mostly used today because of their high efficiency and availability of large amount of data to train the system. The drawback of this approach is the complexity of the system and requirement of large amount of transcribed training data making it unsuited for small scale applications. Therefore approaches based on template matching are still being used in small-scale embedded systems [Dongzhi He, et al. 2010].

# Research Objective

The objective of this research is to develop a simple and efficient speaker independent keyword spotting system using the Template Based approach. Although statistical approaches like HMM have high accuracy, template base methods are still used in small-scale systems like cell phone and mobile application because of their simplicity of the hardware implementation [Zaharia, et al. 2010]. HMMs are known for poorly modeling long temporal dependencies [Grangier, et al. 2007] while template based approach can use the entire temporal context including length and relative position. Techniques like Dynamic Time Warping (DTW) are used to model co-articulation effects or speaker dependencies [Sakoe, et al. 1978]. The major drawback of Template Matching (TM) approaches is that they fail to take the advantage of a large amount of available data. However researchers have worked on TM based approaches that could benefit from training data. In [Zaharia, et al. 2010], the authors have proposed to cluster several templates using quantization algorithms and replace the reference model with centroids in order to achieve a generic template. Researchers are also interested in learning the distance metric that optimizes the recognition performance [Grangier, et al. 2007].

In order to apply DTW to continuous speech, the utterance has to be broken down into isolated words which require knowing the exact beginning and ending of each word. This problem is more prominent for informal conversation where speakers do not follow constrained grammatical constructs and tend to speak at different rates with no distinct boundary between adjacent words. The segmentation algorithm is applied to a continuous stream of speech so that template matching can be performed on isolated words. This is, however, very inefficient and performance of the overall system is tightly coupled with the efficiency of the segmentation algorithm. Another solution to this problem could be to modify the classic boundary-constrained DTW algorithm to have flexible boundaries. This way one does not need to know the exact beginning and end of the word to apply DTW matching. There are number of possible methods to incorporate this feature as discussed in , [Anguera, et al. 2010]. The research presented here aims at implementing unconstrained DTW in order to find the keyword uttered in a continuous utterance. It is assumed that one or more keyword templates are available. The research goal is to utilize those templates and search for those keywords in a given utterance.

The primary database used for this research is Call Home database which contains more than 40 telephone conversations between male and female speakers. The conversations are 30 minutes long and contain varying amounts of background as well as channel noise making the task more challenging. Once the system was designed for the Call Home database and parameters were tuned to achieve with satisfactory performance, it was tested on the Switchboard database. Switchboard database is a collection of about 2,400 two-sided telephone conversations among 543 speakers from all areas of the United States originally collected by Texas Instruments in 1990-1991 under DARPA sponsorship [[http://www.ldc.upenn.edu](http://www.ldc.upenn.edu/)].

# Background Study

Keyword spotting has a rich history in speech processing. Initial experiments were conducted for recognition of isolated keywords (keywords separated by silence) [Sakoe, et al. 1978] followed by identifying connected numbers and voice dialing [Myers, et al. 1980]. The template Based approach is most commonly used where continuous stream of spoken utterance is scanned for a given template of keyword. These are now replaced by statistical learning approaches like Hidden Markov Models (HMM) and Neural Network (NN). The main advantage of a statistical system is that it can learn from the existing data and given today’s scenario where data can be found in abundance, these systems can perform very accurately. Major drawbacks of Template based approach can be summarized as:

* Each keyword requires one or more templates which limit the lexicon size and require larger memory.
* Temporal variation of speech needs to be accounted for. This requires non-linear alignment of keywords for comparison.
* Word boundaries need to be determined precisely. This has large impact on the overall system performance.

Speech processing in general involves a series of steps. First the analog speech is passed through a low pass filter in order to remove noise and band-limit the signal. Then it is sampled and quantized at some sampling rate with a certain number of bits per sample. For telephone line, speech is sampled at the rate of 8 KHz with 8 bits per sample. Since speech is a non-stationary signal, piecewise operation is performed. Speech is segmented into small overlapping frames. The smaller the frame length, more the resolution in time domain but one tends to lose frequency information while larger frame length decreases temporal information As the frame gets larger, the time resolution decreases whereas the frequency resolution increases. Higher temporal resolution is required for speech recognition and related application while speaker dependent applications need to be frequency sensitive because the characteristic features are the formant frequencies. There has been intensive research on the optimal length of frame size [Paliwal, et al. 2008]. Researchers settled on a frame length of size varying from 10-32ms depending on the application.

## Speech Features

Speech signal are not processed in their original form. Some characteristic features are extracted from the original signal that not only reduces the signal dimension but is more efficient in terms of further processing. Efficiency of various speech features were investigated for several decades [Iyer, et al. 2009]. Some experiments were conducted using combined features [Gopalan, et al. 2009] to study the performance gain. Listed below are some speech features used.

### Linear Prediction Coding (LPC)

LPC tries to model human articulation system using an all-pole filter. Input to the filter is either periodic impulses ***V*** or white noise ***UV*** depending on whether the sound is voiced or unvoiced as shown in Figure 3.1 below. The rate of vibration of the vocal cord, also called the pitch of the sound, is determined by the period of the impulse train ***V***. The amplitude of speech is modeled by gain control ***G***.

*V*

Impulse Train

*S(n)*

*U(n)*

White Noise

H(z)

Speech Signal

*UV*

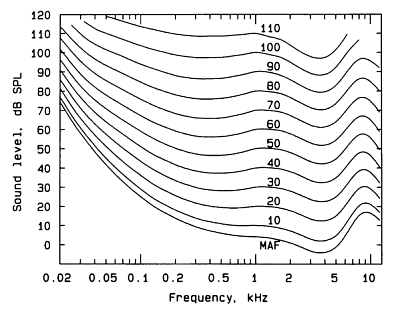
Gain (G)

Equation 3.1

Figure 3.1. Speech generation approximation based on LPC.

The coefficients of the LPC filter ***H(z)*** model the vocal tract. The higher the order of filter, the more accurate is the approximation. Theoretically, nasal sounds require different and more complicated algorithm but in practice they are often approximated by higher order all pole filter.  
  
Using the LPC model, the entire speech signal can be characterized and reproduced if the filter coefficients are computed efficiently. LPC coefficients are mostly used in speech synthesis but are not so effective in speech recognition because they fail to model human perception.

Human perception is non-linear, i.e., the human ear is more sensitive to some frequencies while less sensitive to others. Figure 3.2 below shows the characteristic of human hearing.



##### Figure 3.2. Equal Loudness Curves (also called the Fletcher–Munson curves).

Each curve in Figure 3.2 is perceived equally loud by the human ear despite difference in the amplitude level at different frequencies. For instance, a 30db sound at around 100Hz is perceived as equally loud as 40db sound at 200 Hz or 20db sound at 80Hz. These characteristic can be approximated by scaling different frequencies differently.

**Bark-Scale:**

Bark scale is a psycho-acoustical scale proposed by Eberhard Zwicker in 1961. It was named after Heinrich Barkhausen who proposed the first subjective measurements of loudness [Zwicker. 1961]. The formula that relates the Bark scale to linear scale is given as:

https://lh4.googleusercontent.com/Upf5-Z9MR_tEr7g5e8tGJYBzbkmnaKIe9luIBV3o01XLB8D0rYM5plEELeGMmetOplANo-aY1bBfB71Cd4JAQMh-2Kbx4y1WD3uGI-nARQtaznc_Mg Equation 3.2

**Mel-Scale:**

The name *mel*comes from the word ‘melody’ to indicate that the scale is based on pitch comparisons. This is defined by the relation:

https://lh5.googleusercontent.com/VCHQNykTxRnBLvDXpF4HFE6NTCtOTVJoEIkQVFEb3aUwqk0guXxhLTO65u0v7z3qMpRC-zIm_veCY9VIj6AybzrpBNL6zXI-IlRAyAh6xjd-OGkF7g Equation 3.3

Where m is the mel scale frequency corresponding to linear frequency f.

The Mel scale and the Bark scale can be shown in single plot for comparison.



##### Figure 3.3. Normalized Mel and Bark scaled frequency.

Mel scaling has been more efficient in speech recognition and hence most widely used.

### Mel-Frequency Cepstrum Coefficients (MFCC)

MFCC’s are used extensively in speech processing. Mel warping is performed by a filter bank with center frequencies linearly spaced in the mel-scale. The original spectrum is passed through mel filters and cepstral analysis is performed on the mel scaled spectrum to obtain the MFCCs. Cepstrum analysis basically extracts the information about spectral envelope and formants of the speech signal.

MFCCs are commonly derived as follows:

1. Speech is first divided into short frames over the range it can be considered quasi-stationary.
2. Fourier transform of each frame of the signal is obtained.



……….. 

Frequency

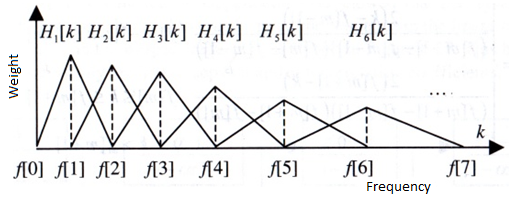
##### Figure 3.4. Speech signal (above) and framwise FFT Magnitude (below).

*In Figure 3.4, in the upper figure, the speech utterance is broken down into frames of 20ms overlapping by 10ms. In the lower figure, FFT magnitude for each frame of the utterance is shown. Thus a continuous 1-D utterance is broken down into smaller 2-D feature (FFT magnitude in this case).*

1. Mel scaling is applied using triangular overlapping windows. There are many variations in the number of and kind of of filter bank used.



3.5. (a)



3.5. (b)

##### Figure 3.5. Different implementations of Mel filter banks.

*As seen in Figure 3.5, the filter bank is comprised of linearly-spaced centers at lower frequency and log- spaced filters at higher frequency. The base of each triangular filter is determined by the center frequencies of the neighboring filters. There are some variations regarding the weights of the filters. Some implementation use filters of unity height while others use filter of linearly decreasing height from 1 to 0.5 as shown in Figure 3.5.a and 3.5.b respectively.*

1. Take the log of the power at each of the mel frequencies.
2. Take the discrete cosine transform (DCT) of the mel log power. These mel-scaled scaled cepstrum exist in a domain referred as *quefrency* which has the same unit as time.
3. The MFCCs are the amplitudes of the resulting spectrum.

Experiments were performed to determine optimal number of filters required. 13 mel-spaced filters are used in today’s state of art system along with ∆ and ∆2 coefficients which are the first and second order derivative of the cepstrum making total of 39 coefficients. Including the ∆ coefficients improves recognition particularly in noisy medium. Some other parameters like log energy, pitch, etc, are also used in conjugation with MFCC features.

### Other Features

Other commonly used features for speech recognition are Perceptual Linear Predictive Coefficients (PLP) [Hermansky, et al. 1992]. PLP modifies the short-term spectrum of speech by several psychophysical based transformations and then takes the LPC of the resulting speech. It is basically a combination of Linear Prediction (LP) and Discrete Fourier Transform (DFT). Features based on Wavelet transformation are also an active area of research but results in no significant advantage over other features. MFCC features along with ∆ and ∆2 coefficients are the ones used in state of art recognizers these days.

Once a set of feature vectors are obtained, they are passed to a comparator. A comparator can be a simple distance calculator or a complex Neural Network system. In the early days of speech recognition, comparators were simple distance calculators. There are number of ways distance between features can be calculated, eg. Euclidean distance, Mahalanobis distance, Kullback-Leibler distance, etc. Researchers were interested to find the best distance measurement for speech and speaker identification. It has been shown that different distances yield different results depending on the feature set used [Iyer, et al. 2009], [Hermansky, et al. 1992]. In the proposed algorithm, each cluster is represented by a mean and all other information about data distribution is discarded. Therefore Euclidean distance is the most appropriate distance measure.

## Common Approaches to Keyword Spotting

Different speakers pronounce a word differently. There can be some variation for a word even from the same speaker. Variation can be in the speed of pronouncing, stress point, depending on the context of the conversation or position of the word in the sentence. Articulation variation is more noticeable in informal conversation when speakers are not bound to follow specific grammatical constraints or choice of words. So, in order to identify if any keyword matches a given reference keyword, these variations need to be accounted. The earliest approach to address these variations efficiently was the dynamic programming approach called Time Normalization or Dynamic Time Warping (DTW) [Sakoe, et al. 1978]. Various other techniques have also been implemented for this problem. Some of them are discussed in the next section.

### Dynamic Time Warping (DTW)

Dynamic time warping is an efficient algorithm to find a non-linear alignment path between two sequences that optimizes their distance. It is obtained by time scaling one of the signals non-linearly so that it aligns with the other. It is an extremely efficient time-series similarity measure. It minimizes the effects of shifting and distortion in signals allowing elastic transformation in time. It is one of the earliest approaches to isolated word recognition. For keyword spotting, a prototype of the keyword is stored as a template and compared to each word in the incoming speech utterance.

##### 

##### Figure 3.6. Dynamic Time Warping.

As shown in Figure 3.6 above, the feature vector for the reference keyword and input keyword are arranged along the two sides of the grid. In this case, the reference template of length *n* is arranged along the horizontal axis and the test word of length *m* along the vertical axis. Each block in the grid is the distance between corresponding feature vectors. The best match between these two sequences can be computed from the path through the grid which minimizes the total cumulative distance between them as shown by the dark line in Figure 3.6. Total distance between the test and the reference data is the sum of distance along the path.

From Figure 3.6 above, it is apparent that the number of possible paths through the grid grows exponentially with the length of the word. Applying some constraints [Sakoe, et al. 1978], the possible paths can be limited to a certain limit making the computation feasible. These constraints basically limit the possibility of infinite paths making computation more efficient. These constraints are:

* Monotonic condition: The path has to be monotonically increasing function and cannot turn back. If i and j are the indices denoting reference and test features, then both i and j either stay the same or increase.
* Continuity condition: There can be no break in the path. i and j can only increase by 1 on each step.
* Adjustment window condition: The optimal path should not wander far away from the diagonal. This constraint is based on the fact that the ideal path is along the diagonal. The maximum deviation is called the window length.
* Boundary condition: The path should start at the bottom left and end at top right. This is logical if the aim is to compare two complete words.
* Slope constraint condition: The path should neither be too steep not too shallow. Too steep of shallow gradient causes an unrealistic correspondence between a very short pattern and a relatively long pattern.

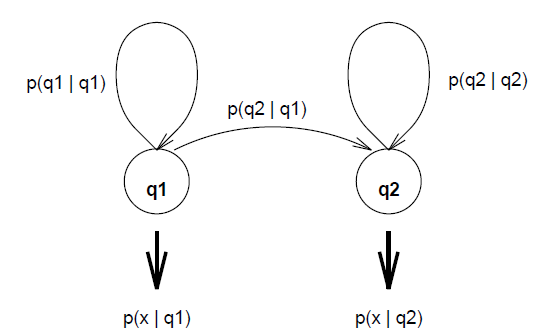
There are different variations to these constraints depending on the application of the algorithm. For connected word recognition, where it is hard to precisely indicate the word boundary, afore mentioned boundary condition is changed to cover some range of possible beginnings and endings of a word. Similarly maximum allowed deviation is selected over the most recent optimal path instead of the diagonal to account for more variation [Myers, et al. 1980].

In order to improve the accuracy of the recognition system, more than one template of same word can be used [Zaharia, et al. 2010]. These systems require more memory in order to store more templates. A more feasible solution to this problem is to develop one generic keyword template in place of many specific ones. The generic template, so developed, has to be speaker independent and should allow some flexibility over speed variation as well as other kinds of articulation variations. Researchers have tried merging all templates by using the mean of feature vectors for every frame and achieve a more generic template using the mean instead of many specific templates [Zaharia, et al. 2010, Zaharia, et al. 2010].

Other researches on DTW based methods include the use of Neural Networks to learn the best distance score that maximizes the recognition performance of system [Grangier, et al. 2007]. A modified version of DTW called the Unconditional-DTW, is being applied to continuous speech where starting and ending of words are not identified [Anguera, et al. 2010], [Park, et al. 2008]. U-DTW approaches are also being used in unsupervised pattern discovery as well as unsupervised document classification [Yaodong Zhang, et al. 2009].

### Hidden Markov Models

Hidden Markov Model is a statistical model in which the system is assumed to be a Markov process with unobservable states. The output of each state however is observable and each state has a probability distribution over the possible outcomes. A simple form of the two states HMM is shown in Figure 3.7 below.



##### Figure 3.7. Two state Hidden Markov Model with hidden states q1, q2 and observation x.

HMMs are “hidden” because the state of the model ‘q’ is not observed; rather the output ‘x’ of a stochastic process attached to that state is observed. This is described by a probability distribution P(x|q). The other set of visible probabilities are the state transition probabilities, P(qi|qj). Assuming that the system is first order Markov process, the probability of next state *j* depends only on the current state *i*.

From the observable sequence of outputs, the most likely system can be inferred. The result is a model for the underlying process. Alternatively, given a sequence of outputs, the most likely sequence of states can be inferred as well.

HMM are most widely used for speech recognition as well as keyword spotting. Assuming the speech signal to be piecewise stationary, each word is modeled as a sequence of stationary units with a certain set of acoustic feature parameters. Although there are several linguistic arguments for choosing units such as syllable or semi-syllable, the unit most commonly used is the phoneme. Isolated word recognition using HMM models can be broken down into two steps.

1. For each word in the vocabulary, build an HMM, i.e., estimate the model parameters that optimizes the likelihood of the training set.
2. Observation sequence is obtained for each word to be recognized by feature extraction and likelihood for all possible models is calculated. The word with highest likelihood is selected as a match. This is done using the Viterbi algorithm to optimize the computation.

For large vocabulary continuous speech recognition (LVSCR), syntactical models are also used along with the word model to apply grammatical constraints. This additional linguistic constraint makes the recognition task easier and also increases the performance of the system. The objective is to output the most probable sentence given the acoustic data X. Thus the utterance S for which the probability P(S|X) is maximum is chosen as best sentence. In order to use HMM, the sentence is represented by a particular state sequence, QN, and the probability required is P(QN | X) which is not directly computable. So this probability is re-expressed using Bayes’ Rule as

Equation 3.4

i.e., divide the estimation process into two parts: acoustic modeling, in which the data probability density P(X|QN)/P(X) are estimated; and language modeling in which prior probabilities of state sequences, P(QN) are estimated.

KWS can be directly implemented using Large Vocabulary Continuous Speech Recognizer (LVSCR). But this requires training the model for a large dictionary of words and huge language model which is rather impractical. An alternative approach is to consider the speech signal to be composed to keywords and non-keywords and using garbage or filler models to characterize non-keywords. Some systems add extra models for representing non-speech events such as silence or coughing. Both the keyword model and the garbage model are built from a concatenation of phoneme sub-models. Likelihood of test utterance matching the keyword model gives the confidence measure of detection of the keyword.

### Other Approaches

The HMM approach requires a large amount of supervised training data, i.e., data that are manually segmented and labeled. The HMM based statistical approach favors the most occurring events because they aim at maximizing the likelihood of an event, but the keywords of interested might not be available in abundance. Moreover, HMM based systems are difficult to train in other languages because of the lack of transcribed data [Hazen, et al. 2009]. They are also known for poorly modeling long temporal dependencies, which needs to be circumvented with refined features or adaptation techniques [Grangier, et al. 2007]. In order to overcome these issues, significant effort has been made towards improving template based recognition and discriminative approaches to keyword spotting [Keshet, et al. 2009].

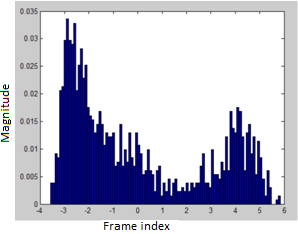
Other methods based on Neural Network are also used for speech recognition. Besides the methods discussed above, there are other methods based on Support Vector Machines SVM [Keshet, et al. 2009], iterative clustering and Self organizing Maps and Learning Vector Quantization [Somervuo, et al. 1999], hybrid methods combining two or more methods are also used for keyword spotting .Besides slight increases of the performance by few points percentage, none of the methods have significantly outperformed original HMM despite tremendous increase in computation.

## Previous Research based on Segmental-DTW

Keyword spotting started off as identifying words in discontinuous speech such as voice dialing [Myers, et al. 1980]. Researchers then started dealing with more complicated scenarios like keyword spotting in continuous speech, meetings and conferences with background noise and cocktail party effect [Bronkhorst. 2000]. The cocktail party effect occurs when conversing in a crowded place with lots of people talking. Most people can ignore the background noise and conversations and concentrate on the person they are talking with. This is possible because the human auditory system can attenuate sound coming from any direction other than the source of interest.

As mentioned in previous sections, there are many approaches to the problem of keyword spotting. The approach in this research is based on a modified version of time warping comparison which is called Segmental Dynamic Time Warping (S-DTW). In this method the test utterance, which is a continuous stream of speech, is first divided into overlapping segments. The starting point of consecutive segment is separated by a certain distance and the warping path is allowed to deviate by certain amount in either side of the center similar to the sliding window technique. Each segment is compared with the keyword template to calculate distortion score. Segments with minimum distortion score are keyword candidates and others with higher distortion score are rejected. There has been some interesting research using this approach which is discussed in the following paragraphs.

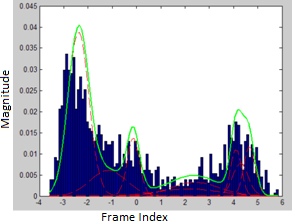
Research has been done on methods based on Gaussian Mixture Models (GMM) [Yaodong Zhang, et al. 2009]. Given one or more instance of keyword to be detected, Gaussian posteriorgrams between keyword(s) and test utterance are compared using Segmental Dynamic Time Warping (SDTW). The decision is made based on the distortion score on the test utterances and best matching candidate are selected as keyword hits. Speech is first divided into short stationary frames. MFCCs features are computed for each frame. This results in a multidimensional MFCC feature vector for each frame. The first row of the feature vector for keyword ‘ONE’ is shown below in Figure 3.8.



##### Figure 3.8. Distribution of the first dimension of MFCC feature vectors for digit ‘ONE’.

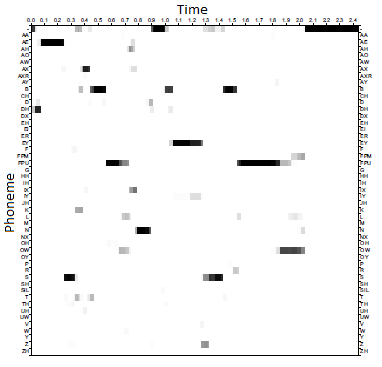
Figure 3.8 shows the distribution of the first dimension of MFCC feature vectors extracted from the training data for the digit ‘one’. A distribution could be used to fit the PDF, but the distribution looks quite arbitrary, and standard distributions do not provide a good fit. One solution is to fit a Gaussian mixture model (GMM), a sum of weighted Gaussians as shown in Figure 3.9.

The complete Gaussian mixture density is parameterized by the mixture weights, mean vectors, and covariance matrices from all component densities. GMM parameters are estimated for entire unlabelled training data initialized by K-means algorithm. This way, the Gaussian Mixture Models (GMMs) are developed for each speech frame for entire training data.



##### Figure 3.9. Approximation of feature vectors using weighted sum of Gaussians. The solid green line shows the best estimate.

For a given keyword, the MFCC feature vector is found in similar manner. The objective is to find a Gaussian posteriorgram distribution for the GMM. Given GMM models and a test feature vector, the log-likelihood value of that feature set, belonging to a particular GMM, is given by the posterior probability. Posterior features have been widely used in template-based systems. The Gaussian posteriorgram is a probability vector representing the posterior probabilities of a set of Gaussian components for a speech frame.



##### Figure 3.10. A grayscale representation of phonetic posteriorgram representation for the spoken phrase ‘basketball and baseball’.

A sample of phonetic posteriorgram is shown in Figure 3.10 for illustration of Gaussian Posteriorgram where the horizontal axis represents time in seconds and the vertical axis represents individual phonetic class. The darkness represent the posterior probability of each frame belonging to a phonetic class, 1 being black and 0 being white. This way phonetic posteriorgrams can be computed for entire utterance.

The difference between the Phonetic Posteriorgram method and the Gaussian Posteriorgrams method is that in phonetic posteriorgrams, probability of a frame belonging to particular phoneme class comprises the posteriorgrams while in Gaussian Posteriorgrams, the probability of a frame belonging to particular Gaussian Model comprises the posteriorgram. If a speech utterance with n frames is denoted as S = (s1,s2,…,sn), then the Gaussian posteriorgram (GP) is defined as:

GP(S) = (q1,q2,…qn) Equation 3.5

Each qi vector can be calculated by

qi = ( P(C1|Si), P(C2|si), …. , P(Cm|si) ) Equation 3.6

Where Ci represents the ith Gaussian component of a GMM and m denotes the number of Gaussian components.

The Gaussian Posteriorgram approach differs from the Phonetic Posteriorgram approach in that the GMM is computed over unsupervised training data instead of using a phonetic recognizer [Hazen, et al. 2009]. As a result, the posteriorgram obtained becomes a Gaussian posteriorgram instead of phonetic posteriorgram. Then segmental time warping is applied to compare the Gaussian posteriorgrams between keyword and test utterance to identify a possible match. First a GMM is trained over unsupervised training data to produce a raw posteriorgram vector for each speech frame.

Since the training is unsupervised, induction of noise in the training data can easily generate an unbalanced GMM because of their large variance. This results in posteriorgram that cannot discriminate well between phonetic units. In order to resolve this issue, a speech detector is used to extract only speech and train the GMM on speech segments only. To avoid approximation errors, a probability floor threshold is applied setting posterior probabilities less than Pmin to 0. The vector is then renormalized to set summation of each dimension to 1. This would create many zeros in the posteriorgram. In order to avoid that, a discounting based smoothing strategy is applied to move a small portion of probability mass from non-zero dimension to zero dimensions. For each Gaussian posteriorgram vector q, each zero dimension *zi* is assigned to *zi = λ/ Count(z)* where *Count(z)* denotes the number of zero dimensions. Each non-zero dimension vi is changed to *vi = (1- λ)vi*.

The difference between two Gaussian posterior vectors *p* and *q* is defined as:

D(*p*,*q*) = - log (*p* . *q*) Equation 3.7

Since both p and q are probability vectors, the dot product provides the probability of these two vectors drawing from the same underlying distribution. Dynamic Time Warping algorithm is later used to scan through the posterior probability to identify possible keywords.

# Initial Research

Before discussing the speech recognition, a basic understanding of speech characteristics is essential. The first sets of experiments were related to studying speech characteristics by applying basic signal processing algorithms to speech. Since the basic tool for measuring the similarity of two signals differing in time is cross-correlation, it seems logical to start the research by cross-correlating the keyword with the utterance.

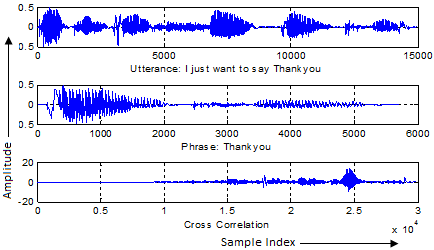
## Cross-Correlation

Cross-correlation is a measure of similarity of two waveforms as a function of a time-lag applied to one of them. The cross-correlation coefficient should be higher at the keyword and less elsewhere. First set of experiments consisted of short utterance, 1-3 words long. In order to estimate the result, each keyword was cross-correlated with the portion of utterance containing the keyword. A peak could be seen at the location of the keyword as shown in Figure 4.1 below. This way one can not only identify the keyword but also estimate its position in the utterance.

##### 

##### Figure 4.1. Cross-correlation (bottom) of utterance ‘I JUST WANT TO SAY THANK YOU’ (top) and ‘THANK YOU’ (middle) portion of the same (sample 9500-15000 of the utterance marked by the circle).

Next, the experiment was repeated for different instance of the same keyword spoken by same person.



##### Figure 4.2. Cross-correlation (bottom) of same utterance (top) and different instance of phrase ‘THANK YOU’ (middle). The circle highlights the keyword portion in the utterance.

In Figure 4.2 above, high energy at the region of keyword can be seen with a peak at the center. Additionally, there is monotonous increase and decrease in amplitude around the peak indicating high probability of keyword at that location. But articulation varies not only with speakers but also for the same speaker. But in other cases, cross-correlation failed to produce high energy at the location of the keyword.

##### 

##### Figure 4.3. Correlation between the utterance ‘THEY JUST RECORDING OUR CONVERSATION THEY USE IT LIKE’ and keyword ‘CONVERSATION’ spoken by same person. The circle highlights keyword poriton in the utterance.

As seen in Figure 4.3, the correlation result is low indicating absence of the keyword ‘CONVERSATION’ in the utterance.



Figure 4.4. Different instance of keyword ‘REALLY’.

In order to investigate the reason behind this, variation in different keywords were investigated comparing different instance of each. Figure 4.4 shows different waveforms for the word REALLY spoken by different speakers. It can be seen that although some of them look quite similar (1, 3 and 6; 13 and 14), there is a lot of variation in the temporal content (4, 5, 15) and also in the length of the keyword (4 - around 1500 samples, 14 - 4100 samples). From Figure 4.4, it can be concluded that analysis in time domain cannot sufficiently describe the speech. For this reason, frequency domain analysis is widely used in speech processing.

## Short-Time Fourier Transform (STFT) Analysis

In order to analyze speech in the frequency domain, it has to be noted that speech is a non-stationary signal. Therefore, piecewise stationary assumption over a small time-frame of 10-30ms is common in speech processing. So the speech utterance is divided into small frames and each frame is analyzed in the transform domain. For the experiment, hamming window of 20ms was chosen for segmentation with an overlap of 10ms.



##### Fiugre 4.5.a. STFT analysis for a portion of two instance of keyword REALLY.



##### Figure 4.5.b. STFT comparison for portions of two instances of the words REALLY.

Figure 4.5 is plot of the magnitude of Short Time Fourier Transform for a portion of keyword REALLY. Figure 4.5.a is the result for same speaker while Figure 4.5.b is for different speakers. It can be seen that FFT for same speaker are similar but there is lot of variation in the case of different speakers. From this figure two major conclusions can be derived:

* One cannot expect a 100% match between keywords even for the same speaker.
* The difference in rate of speaking needs to be accounted for which is often the case even for the same speaker.

## Dynamic Time Warping on LPCC features

One way to overcome the alignment variation is to apply Dynamic Time Warping. This would non-linearly scale the test utterance to match the reference utteranc. The distance along the warped path however will not be zero because of some articulation variation. To account for length variation, the distance is normalized by dividing by length of the utterance.

To verify this concept, experiments were performed computing the distance between same and different keywords. Each keyword was divided into frames of 20ms with an overlap of 10 ms using a hamming window. 13 LPC coefficients were derived for each frame and Euclidean distance between the features was summed up along the warping path as a distance score for the keyword.

##### Figure 4.6. Distance between various keywords computed along warped path.

Figure 4.6 shows the result of warped distance calculation between four different keywords. From the results above, the following observations can be made.

* Distance between same keyword is less than between different keywords. It can be assumed that distance between same keywords, in general, is less than that between non-keywords, silence or noise.
* If a threshold is to be determined to differentiate between keyword and non-keywords, it is going to be different for different keywords. It can be seen from the graphs that the four keywords have require different vale for threshold. Therefore one cannot expect to achieve universal threshold for all keywords.

###### Table 4.1. Different threshold for different keywords.

|  |  |
| --- | --- |
| Keyword | Threshold |
| Conversation | 1.5 |
| Bizarre | 1.8 |
| Something | 1.0 |
| Relationship | 1.4 |

* Discrimination would be easier for some keywords because of large variance from other keywords like SOMETHING in the results shown in Figure 4.6. Therefore it is expectable to have different results for different keywords categorizing the keywords as GOOD keywords and BAD keywords.

Threshold obtained from the experiments were used for keyword spotting in continuous speech. Since length of keyword in the utterance was unknown, the average length of all template keywords Lavg was selected as hypothetical keyword length and the utterance was divided into segments of length Lavg with overlap varying from Lavg/2 to Lavg/8. Euclidean distance was summed along the warped path and normalized by dividing by path length as match score.

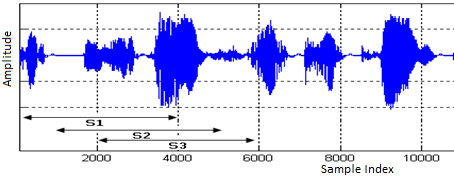


Figure 4.7. Utterance divided into overlapping segments s1, s2, etc. of length Lavg.

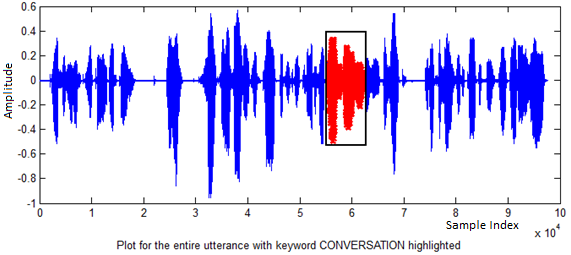


Figure 4.8.a.

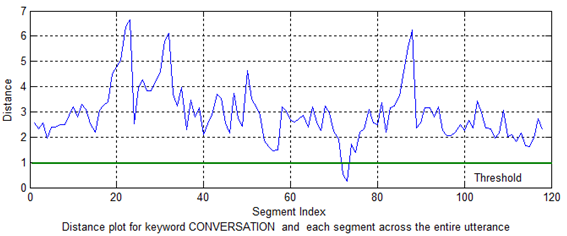


Figure 4.8.b.

##### Figure 4.8. Keyword detection using a threshold value. Figure 4.8.a shows segment of the utterance with keyword CONVERSATION highlighted on red and Figure 4.8.b. shows the segmental distance with threshold indicated by the green horizontal line.

Figure 4.8 is plot of distance between the keyword and the segmented utterance along the utterance. Variation of distance for every frame along the utterance can be seen in the figure. Assuming a predefined threshold as shown by the green horizontal line in the plot below in Figure 4.8, the keyword is spotted when distance is below the threshold as seen above around frame 75.

The algorithm worked for some utterances but detection rate was on average below 50%. There were many misses as well as false alarms. This indicates that there was some problem in the algorithm. A possible explanation could be that the length of keyword in the utterance differs from the average length calculated. In such case, the comparison is between template keyword against test-keyword with additional noise or some missing portion. Moreover dynamic time warping requires the knowledge of exact beginning and end of the keyword in the utterance. Therefore in order to achieve good performance, keyword boundaries need to be identified which is a big challenge in itself. Even manual labeling of the word boundary is prone to human error. Keeping account of these limitations, it was concluded that the algorithm had to be revised.

Experimental result on continuous speech is shown below in Figure 4.9 for 13th order MFCC feature vectors. Shown in Figure 4.9.a. is a two dimensional distance matrix of size Nk\*Ns,where Nk is the number of frames in test keyword and Ns is the number of frames in the speech segment. If the keyword is present in the utterance, a monotonous minimal distance path has to be present indicated by dark straight line starting from the beginning of the keyword. Figure 4.9.b. below is distance plot of keyword vs. utterance in terms of intensity of each pixel, darker pixel means smaller distance while brighter intensity represent greater distance.



Figure 4.9.a. Grayscale Distance plot of keyword Vs Utterance.



Figure 4.9.b. Enlarged portion between frames 400-540 of Figure 4.9.a.

##### Figure 4.9. Euclidean Distance plot for entire utterance and the keyword CONVERSATION in grayscale.

The goal is to identify the diagonal minimal-distance path (dark line) as seen in Figure 4.9.b (between frame 400-540), as it would not only indicate the presence of keyword but also provide its location.

## Keyword Statistics

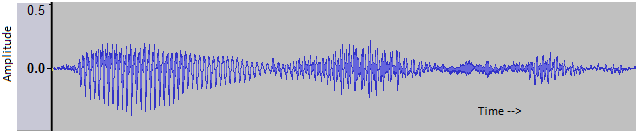
Keyword spotting techniques are not fully keyword independent and the choice of keywords influences the performance of the algorithm [Silaghi. 2008]. Certain words are recognized more easily than others. So the system has to be tested on various kinds of keywords in order to estimate is overall accuracy. For this, various keywords were chosen with different length and phoneme count. Shown below in Table 4.1 are parameters of some of the keywords studied.

###### Table 4.2. List of keywords the system was tested on.

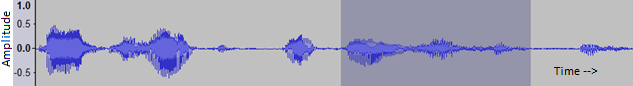
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Keyword | No. of Syllables | No. of Phonemes | Average Length (secs) | Length Variance |
| University | 5 | 10 | 0.62 | 0.09 |
| Conversation | 4 | 10 | 0.66 | 0.18 |
| Computer | 3 | 8 | 0.38 | 0.08 |
| College | 2 | 5 | 0.45 | 0.09 |
| English | 2 | 6 | 0.39 | 0.07 |
| Lanuage | 2 | 7 | 0.44 | 0.10 |
| Program | 2 | 7 | 0.55 | 0.13 |
| School | 1 | 4 | 0.40 | 0.09 |
| Something | 2 | 6 | 0.42 | 0.11 |
| Student | 2 | 7 | 0.37 | 0.11 |
| Bizarre | 2 | 5 | 0.52 | 0.15 |
| Circumstance | 3 | 10 | 0.58 | 0.02 |
| Necessarily | 5 | 10 | 0.61 | 0.13 |

[\* The phoneme and syllabus count are based on <http://www.howmanysyllables.com/>, <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>]

In continuous conversation, keywords are not pronounced as isolated words. Isolating the keyword from the utterance is a difficult task as it does not have distinct boundary and the first syllable of the keyword is usually mixed with the last of the preceding word with inseparable transition. In some cases, the keyword, isolated from continuous stream of speech, vary from the original keyword. Even a human listener cannot identify the keyword after segmentation. But when listening with the context, it can easily be identified.



##### Figure 4.10. Utterance segment that sounds like ‘KINGWISHA’.



##### Figure 4.11. Utterance ‘You have to speak English for some’. The keyword ‘ENGLISH’ is distinct here when heard with the context.

Figure 4.10 is the portion of utterance ‘YOU HAVE TO SPEAK ENGLISH FOR SOME’ containing the portion ENGLISH which sounds more like KINGWISHA due to the overlap of preceding ‘SPEAK’ and following ‘FOR’ with keyword ‘ENGLISH’. The keyword does not usually appear as a sequence of precise syllable or phonemes in an utterance and this is a challenge when designing a keyword spotting system.



# Keyword Spotting System Design

## System Overview

Any general speech recognition system can be divided into two major operations, Training and Testing. The proposed approach is based on a time warped comparison of quantized features of the test keyword and the utterance. In order to quantize the speech feature, a feature space is generated that is composed of all possible sound variations. The space is then quantized such that each section in the cluster represents a unique sound. This is used to discriminate between different sounds. A keyword can be considered as a sequence of different sounds, a template matching algorithm can be used to measure similarity in the sequence between the template keyword and the test utterance. It is easier to visualize the cluster if each section is assumed to represent a syllable or a phoneme as in Figure 5.1.

##### Figure 5.1. Phoneme Cluster. Feature space divided into three phonemes /ER/, /SH/, /AA/. There is no clear separation at the boundary condition and a feature belonging to one class might fall into other.

Using the cluster shown in Figure 5.1, a keyword can be identified in an utterance by detecting if same sequence of phoneme is present in the utterance. Keyword spotting using template matching approach on phonetic posteriorgrams has been studied . For this, every keyword and utterance has first to be phonetically labeled. The training data has to be phonetically transcribed with high accuracy which requires manual transcription. This is not practically feasible especially for languages other than English. Commercial speech recognizers have limited language coverage (50-100 languages) which falls dramatically short of covering nearly 7,000 languages spoken all around the world [Yaodong Zhang, et al. 2009]. In order to cover those many languages, an unsupervised cluster training method is required. Moreover, for keyword detection, the utterance needs to be converted into a phonetic string before template matching can be performed which requires complicated methods like HMM. The accuracy of the detection algorithm is dependent on the accuracy of the phoneme detection algorithm thereby reducing the overall accuracy.

Research has been performed on methods based on Gaussian Mixture Models (GMM) [Yaodong Zhang, et al. 2009][Yaodong Zhang, et al. 2009]. The cluster is composed of Gaussian Models and the probability of a vector belonging to a particular cluster is given by the Gaussian posteriorgram vectors. The difference between posterior vectors of keyword and utterance is given by their dot product as D(p,q) = -log(p . q) where p and q belong to the keyword and utterance respectively. This method does not require transcribing the training data. Gaussian Mixture Models are computed for the entire training data and keyword detection is based on posterior probability measure along a non-linear path. Computing Gaussian Mixture Models, posterior probabilities and the dot product for every frame is costly. Therefore a simpler method is implemented in this research.

In this research, the cluster is composed of speech vectors partitioned by K-means clustering algorithm. Each cluster is represented by a single centroid without any other information about data distribution within the cluster. GMMs are composed of means and variances whereas the clustering method used in this research is represented by a single mean. This reduces the computational complexity of the training process. Likelihood of a speech frame belonging to a cluster class is given by the Euclidean distance between them which is far more computationally efficient than computing posterior probabilities. Euclidean distance also gives the distance between a keyword frame and a speech frame which is easier to compute than the log of dot product of the two posterior probabilities.

## System Training



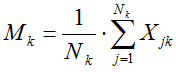
### Cluster Training

A keyword can be considered as a string of sounds. Detecting a keyword in an utterance can be thought of as searching for the short string in a long segment. It is important to keep in mind that the same keyword even from the same speaker does not necessarily sound identical. This leads to variation in the extracted speech features. In order to account for those small variations, quantization is done on the speech features dividing the entire feature space into finite clusters. This process is called Vector Quantization.

The basic idea is to populate a feature space that would consist of all possible sounds and split it into a set of finite clusters such that similar sounds belong to same cluster. For this, a feature space is populated from features extracted from long speech utterances from different speakers (i.e., diverse enough to contain almost all possible sounds) as shown in Figure 5.2. Each point in the feature space represents a unique feature vector. Keeping in mind there can be some variation, feature for the same sound might not coincide exactly but could fall in the neighborhood of the first. So the space is divided into a set of finite clusters (64-512 for the experiment) so that each cluster represents a similar sound. Since the goal is to identify a region in the speech utterance that has the same sequence as that in the keyword, the overall recognition is more sensitive to continuity of the patterns than the precision of single cluster.

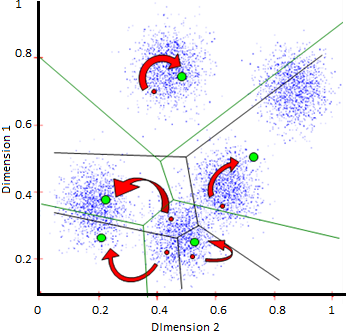
For quantization, K-means clustering algorithm [Lloyd. 1982] is used. The K-Means algorithm tends to minimize the within-cluster sum of distances from the centroid. Given below are the steps for K-means clustering.

* Initialize K cluster means by randomly selecting them from the training set, where K is the number of clusters required.
* Divide the entire training set into clusters by associating every observation to the nearest mean. In other words, for each vector Xi in the training set, the nearest cluster center C\* is found and Xi is associated to the cluster with centorid C\*.
* For each cluster, re-compute its center by finding the mean of the cluster:

 Equation 5.1

Where Mk is the new mean, Nk is the number of vectors in cluster k, and Xjk is the k-th pattern belonging to that cluster.

* Update the centroid with the new means.
* Repeat the process until convergence is reached, i.e., the distance between new and previous mean is minimal (e.g., 1%).



##### Figure 5.2. 2-D Vector Quantization – Feature space quantized into clusters (5) each represented by a centroid. The red and green dots represnts centroids after first and second iterations.

Figure 5.2 is an illustration of a 2-dimensional vector quantization. To start with, K centroids are randomly selected from the cluster shown by red small dots in the figure. Every feature in the space is then associated to the nearest centroid. This divides the feature space into K clusters. The mean feature is computed for each cluster which replaces the old centroid shown by green dots in the figure. The process of mean refinement is performed iteratively untill convergence is achieved.

Each 2-D vector falling in a particular region is approximated by the centroid of the region as shown by colored circles. As a result, the entire space is divided into 5 regions approximated by 5 centroids. Since each centroid has a unique index, the vector sequence can be represented as sequence of cluster indices which can drastically reduce the computation complexity.

Speech features are high dimensional vectors resulting in a high dimensional feature space, but the same algorithm described above can be applied easily. Each frame can then be represented by an index resulting in a codebook of indices for a given frame sequence. This not only reduces the dimension and hence the computation complexity but also eliminates small variations in the feature vectors by quantizing it to nearest centroid.

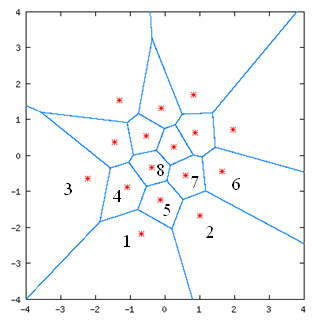
### Codebook Generation

The system is designed on the assumption that one or more instance of the test keyword are available. Once the cluster is trained, the feature space is quantized into K clusters represented by K centroids which are each an N-dimensional feature vector. Every keyword template available is then quantized using this cluster so that each frame is represented by the centroid closest to it, this way, each keyword can be represented by a string of 1 dimensional codebook indices rather than N-dimensional feature vectors. This makes keyword representation simpler and computationally efficient. If more than one keyword template is available, the best way to deal with it would be to average all the features and represent the generic keyword as string of average codebook indices.

The speech utterance is also converted into a one-dimensional string of codebook indices in the same way described above. This results in a short string of indices to represent each template keyword and a longer string representing the utterance.

### Distance Matrix Computation

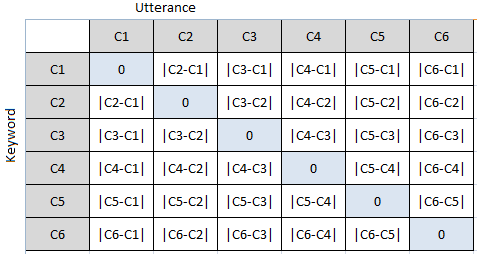
Before going into how to compute a distance matrix, let us first discuss the purpose of the distance matrix. A distance matrix is a likelihood estimate of a centroid belonging to any cluster. A speech frame belonging to a particular cluster with larger variation might fall into an adjacent cluster rather than the cluster it should belong to. So computation of the likelihood of an observation belonging to any other cluster sounds logical. This can be computed by the measure of distance between the observation and other cluster centroids. Applying this method would increase the flexibility allowing an accommodation of comparatively larger variation in articulation.



##### Figure 5.3. Likelihood Estimation by distance measure.

As in Figure 5.3, vectors in region 2, 3, 4 or 5 could more likely to belong to region 1 than to region 6. This can be mathematically estimated by using the distance between centroid of the regions corresponding to the vector. Since the feature vector is already approximated by cluster centorid, the likelihood of a vector belonging to different cluster is approximated by the distance between centroid of the two clusters. For example, vector V1 in cluster R1 (Region 1) is quantized to C1 (centroid of region 1). So the likelihood of it belonging to R2 is proportional to the distance measure |C2-C1|.

For a codebook with 6 clusters (K=6) the distance matrix D looks like:



##### Figure 5.4. Distance matrix for a codebook with 6 clusters.

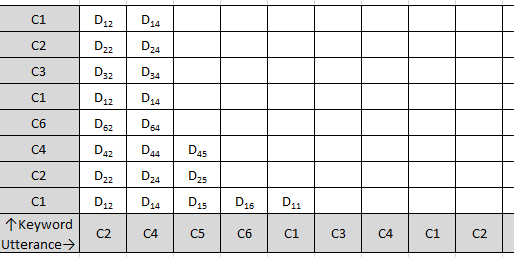
As shown in Figure 5.4, Distance matrix D is a symmetric matrix with all diagonal elements equal to 0 (Dij= 0 for i = j and Dji=Dij for i ≠ j). The reason this matrix is pre-computed is to expedite the detection process. Once the template keyword and the utterance are quantized and converted into string of codebook indices, the likelihood of a frame can be directly estimated using the distance matrix without having to refer to the original cluster.

After vector quantization, keyword and utterance takes the form of codebook, i.e., sequence of cluster indices as shown below.

Keyword: C1-C2-C4-C6-C1-C3-C2-C1

Speech segment: C2-C4-C5-C6-C1-C3-C4-C1-C2-C3-C6

Then the likelihood matrix will be:



##### Figure 5.5. sample likelihood matrix for quantized keyword and utterance.

Figure 5.5 shows the computation of likelihood matrix for quantized keyword and utterance. Each cell in the matrix can be determined from the distance matrix D. For example distance between the first frame in the keyword belonging to cluster C1 and the first frame in the utterance belonging to cluster C2 (left bottom cell in Figure 5.5) is D12 where D12 = |C1-C2|. This avoids a large amount of computation during keyword detection.

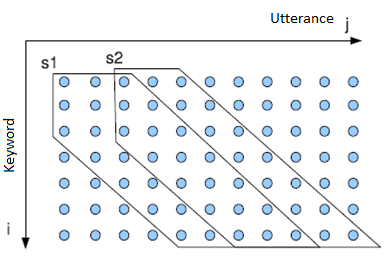
## Keyword Detection

Vector quantization results in a codebook with finite clusters, each represented by single centroid. The template keyword(s) are also available in the form of string of codebook indices, and the distance matrix. For identifying the keyword in an utterance, the utterance is first converted into string of indices in the same way as keyword. This results in template keyword(s) and utterance in the form of 1-D string of codebook indices. If the test keyword is present in the utterance, a similar pattern can be expected in the utterance and the distance plot would be a monotonically decreasing continuous dark line as was shown in Figure 4.9. In order to account for some articulation variation in terms of speed as well as pronunciation, a flexible comparison technique which is called Segmental Dynamic Time Warping is applied.

### Segmental – Dynamic Time Warping

Dynamic time warping is a method to compare two words by time warping for optimal alignment while Segmental DTW takes two continuous utterances and computes matching pair of subsequences. Segmental DTW has been successful in terms of unsupervised pattern discovery [Anguera, et al. 2010], . Recent work suggests that this method can also be applied to unsupervised keyword spotting , [Yaodong Zhang, et al. 2009]. The major modification over traditional DTW for this implementation is the imposition of two constraints. The first one is the adjustment window constraint to restrict the shapes that a warping path can take as already defined in Section 3.2.1. Second, multiple alignment paths are allowed for the same two input sequences by employing different starting and ending points in traditional DTW search algorithm.

The adjustment window constraint introduces a division of the search grid into finite segments for generating multiple alignment paths.



##### Figure 5.6. Segmental DTW for adjustment window size 2.

As shown in Figure 5.6, the adjustment window condition not only restricts the shape of the warping path but also the ending coordinate for a given starting point. For example, if the starting coordinate (i1, j1)is (1,1) and the end iend = m, then jend ϵ (1+m-R, 1+m+R). As a result, the difference matrix can be divided into several continuous diagonal segments of width 2R+1. In order to avoid redundant computation and taking into account warping paths across segmentation boundaries, use of overlapping sliding window seems plausible as indicated by starting points s1 and s2 in Figure 5.6.

Applying Segmental DTW over the test utterance results in (n-1)/R warping paths, where n is the number of frames in the utterance and R is the window size. Each path has its corresponding path indices in terms of i and j, path lengths, distortion score and number of discontinuities for each test utterance. Segments with minimum distortion are selected as candidate region of keyword occurrences. Since the goal is to find portions of the utterance which are similar to the keyword, the next step is to discard the paths with high distortion, low score and larger discontinuities.

### Decision Logic

Once the distortion score is computed for the entire utterance, the detection decision can be made based on a threshold value. Different keyword might have different distortion threshold of detection. Computing a universal threshold can be difficult task because it might not exist. If more than one instance of the keyword is available, an efficient way of taking advantage of the redundancy would be to normalize all the template keywords to obtain a generic template keyword that would fit them all. Theoretically, it can be obtained by modifying the k-means algorithm for vector quantization. For instance, if 3 instances of a keyword are available with following cluster indices,

K1: C1, C5, C20, C53, C200, C150, C10, ….

K2: C2, C6, C20, C52, C198, C150, C9, …..

K3: C1, C6, C22, C50, C200, C148, C8, …..

The goal is to rearrange the cluster so that first frame of all three keywords fall into single cluster and so is represented by a single centroid. Discrimination capability of that cluster for these frames would be much higher than for the normal cluster obtained through k-means algorithm. However some keywords can have higher variation. For instance, for another keyword the codebook is K4: C5, C100, C50, C200, C150, C10, etc. Grouping those into single cluster would induce noise in that cluster reducing the overall discrimination capability. Hence, the modified map should try to include most but not all of the frames. Self Organizing Vector Maps (SOM) [Somervuo, et al. 1999], kernel methods such as Support Vector Machines (SVM) [Keshet, et al. 2009] could be used for this kind problem which is outside the scope of this research.

Another way to take advantage of the availability of multiple keyword templates would be to obtain a segmental score for each keyword and merge those to obtain the final score using some kind of fusion algorithm. A segment with high average score will most likely be the keyword. This method is computationally expensive but easier to implement compared to the first approach of modeling all available keyword templates into single generic template.

A voting based approach could also be used as decision logic. If many of the available templates detect the keyword at same segment location, the likelihood of that segment being the keyword is high. However if only a few template give positive detection at some location, the likelihood is low. Voting based approach is the one currently implemented in proposed system because of its efficiency and computational simplicity.

## Experimental Setup

### Corpus

Experiments were performed initially on the Call-Home database. The system was later tested on Switchboard database. The Call-Home database is a collection of thirty minutes recording of conversation between two speakers (same or different gender) over telephone line. The conversations were first split into single channel (mono) for speaker separation. Since the conversations were real time, one speaker waits while the other is speaking. This results in long pauses in single speaker’s speech. Long silence in the speech would result in a large number of near zero speech features which can result in unbalanced cluster if not removed from the training data. In order to avoid those pauses, a simple energy based speech detection block was added which eliminates long silences in the data. Since it is required that the training feature space is populated with enough data from various speakers, 5 minutes of speech (without pauses) was taken from different speakers (14 speakers) resulting in 5\*14 = 70 minutes of training data.

### Pre-Emphasis

Speech signals have a spectrum that falls off at high frequencies. In some applications it is desirable that this high-frequency fall-off be compensated by pre-emphasis. This process therefore serves to flatten the signal so that the resulting spectrum consists of formants of similar heights. Formants are the highly visible resonances or peaks in the spectrum of the speech signal, where most of the energy is concentrated. The flatter spectrum allows more accurate speech analysis because without it the extracted features would be highly dependent on lower formants losing important information about certain sounds. A simple method of pre-emphasis is processing with a first order high pass filter given by:

S2[n] = s[n] - αs[n - 1] Equation 5.2

The Z-Transform of the filter is given by:

H[Z]=1- α Z-1  Equation 5.3

Where, s[n] is the input speech signal, S2[n] is the output pre-emphasized speech and α is the adjustment parameter. The closer α gets to 1, the more the high-frequency is emphasized. Typical values for α range from 0.9 to 0.95 for speech processing. For this experiment α was set to 0.95.

### Feature Extraction

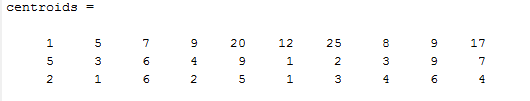
Speech features (13 MFCCs, 13 ∆-MFCCs and 13 ∆2-MFCCs resulting in 39 feature coefficients) were extracted from the training data after pre-emphasis. MFCC features were computed using filter-banks with 24 filters. Speech data was divided into frames of 25 ms (200 samples) with 50% overlap (100 samples) using a Hamming window. This setup was based on experimental results and results for 26 coefficients (excluding ∆2 coefficients) are also presented.

### Quantization

Features obtained were then quantized using the K-means clustering algorithm. This resulted in specific set of quantized feature vectors. Different values of k were chosen in order to study the effect of cluster size on the system.

After quantization, an index is assigned to each cluster resulting in a codebook with k indices. In order to speed up later computation, the distance matrix of features in the codebook was pre-computed resulting in a k-by-k distance matrix.

For example, assuming 3rd order feature vectors extracted from a short speech segment, quantization into 10 clusters (k=10) results in 10 centroids (3\*10) which is shown in Figure 5.7.



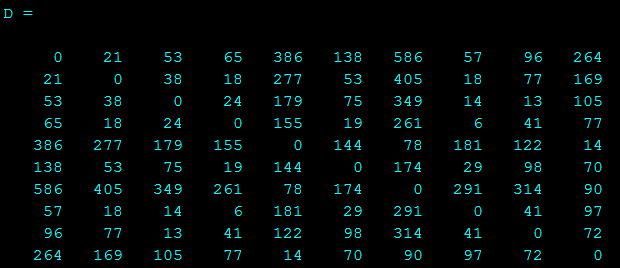
##### Figure 5.7. Sample Cluster centroids for cluster size k=10.

Assigning an index to each centroid vector will result in a 1-dimensional codebook of size 10 (say C 1-C10). The distance matrix contains distance between each of the elements which would be a 10-by-10 matrix that is shown in Figure 5.8 below.

D[1][1] = |1-1|2 + |5-5|2 + |2-2|2 = 0

D[1][2] = |1-5|2 + |5-3|2 + |2-1|2 = 21 = D[2][1]

D[1][3] = |1-7|2 + |5-6|2 + |2-6|2 = 53 = D[2][1] ...



##### Figure 5.8. Distance matrix for cluster shown in Figure 5.7.

It can be noted that all the diagonal elements are 0. The purpose of computing this distance matrix is to make later computation faster. From this matrix, it can be easily seen that the distance between 5th and 9th element of the codebook is D[5][9] = D[9][5] = 122. So it is not required to look back to the centroid vectors and compute distance again.

### Keyword Template generation

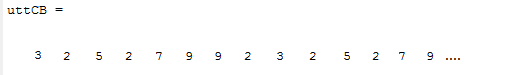
Once the cluster centroid is obtained from the training data, speech features from each frame in the keyword(s) is associated to a centroid. The actual vector information is then discarded and only the codebook index for each frame is retained. The process is repeated for each instance of keyword available which results in 1-D string of indices representation for each test keyword. A test keyword sample is shown below.



Where, 5,3,2,… are the indices of feature codebook.

### Utterance Processing

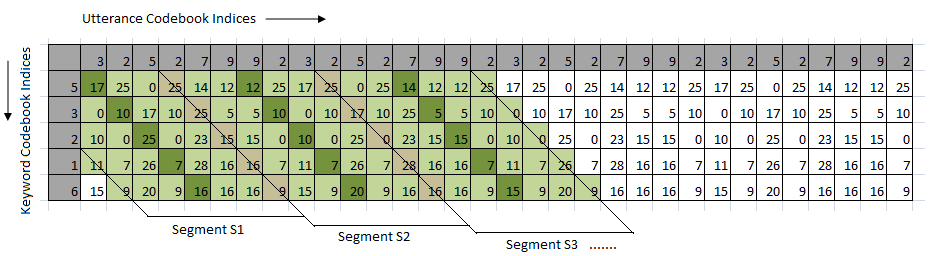
The test utterance is processed in the same way as template keywords. This results in a long string of codebook indices similar to one for each keyword template. A sample utterance codebook is shown below.



### Likelihood Comparison

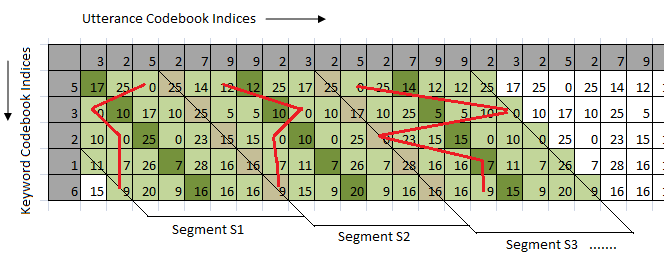
To compute the likelihood of a keyword being in the utterance, the utterance is divided into overlapping segments as described in Section 5.2.1. Every segment is then matched against each of the template keywords. In order to account for the possible variation in spoken word, the time warped distance was computed between keyword and each segment rather than using the linear distance.

After the keyword and utterance codebook are obtained, they are arranged along two sides of a grid. For illustration, let us consider previous sample utterance and keyword. From the distance matrix, distance between corresponding codebook indices can be obtained directly as shown in Figure 5.9 below.



##### Figure 5.9. Frame-wise distance between keyword and utterance divided into segments. The dark green blocks represent the center of each segment separated by dark line.

The darker blocks represent the center of each segment. It can be seen in Figure 5.9 that the segment width is seven units and there is an overlap of one unit on both side of every segment except for the first and the last segment. For the likelihood computation, the distance of minimum distortion path is cumulated and divided by the length of the keyword codebook as shown in Figure 5.10.



##### Figure 5.10. Minimal distance path for each segment shown by red line. Total distance is the sum of distance along these path divided by keyword length.

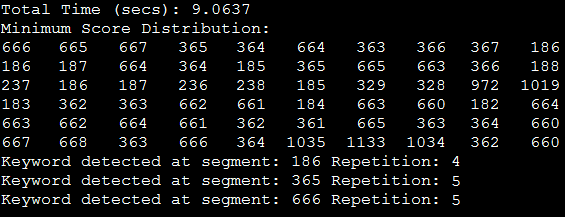
For instance,

Distortion score for segment S1 = 0+0+0+7+9/5 = 2.4

Distortion score for segment S2 = 12+0+0+7+9/5 = 5.6

Each segment has a likelihood score. The high scorer segments (low distortion segments) are the candidate keywords in the utterance. In order to take advantage of the availability of multiple keyword templates, voting based approach is used for decision logic as described in Section 5.3.2. The top ten scorer segments are selected for each keyword and the common segments are counted for all available keyword templates. If the repetition is greater than N/2 (N being the number of keyword templates available), the segment is identified as the keyword or else it is rejected. In other words, if half the keyword templates indicate there is keyword present at certain location of the utterance, it must be the keyword.

Shown below in Figure 5.11 is an illustration of how decision logic is implemented.



##### Figure 5.11. Illustration of Keyword detection by voting based approach. If a segment is detected for more than half the keyword templates (n ≥ N/2, N = 6), it is considered a keyword.

Figure 5.11 is an illustration of decision logic implemented using voting based approach for six keyword templates. The utterance length is 20 seconds which is divided into total of 1222 overlapping segments. The execution time is less than half the utterance length (9.06 secs). The top ten lowest distortion segments are selected for each keyword templates. Then frequency of occurrence for the segment is counted in other keyword template. The segment 666 is the top scorer for first template. It is to be noted that not all keyword templates are identical. So there can be some deviation in the exact segment location in the utterance. So the occurrence of segments near 666 (666 ± 5) is counted in the other keyword templates. It can be seen that segment 664 occurs at third position for second keyword template. This adds up the vote for segment 666 to 2. Continuing in the same way, the total votes for segment 666 cumulates to 5 which is greater than 3. Therefore, segment 666 is considered as a keyword. Similarly segment 186 gets 4 votes and 365 gets 5 votes resulting in 3 detected keywords.

In order to express the result in terms of time, the detected segment is divided by total number of segments and multiplied by the time length of the utterance. For example, the time location of segment 666 is:

Keyword Location = 666/1222 \* 19.506 = 10.63 secs

The actual location of keyword in the utterance is at 10.518 sec with length of 0.689 secs. Therefore the precision error of the detected keyword is around 17%. Precision error of greater than 30% is considered misdetection or a false alarm.

# Results

Preliminary experiments were performed to determine how individual templates would behave and what the average score would be. Given below are some results for different keywords, different cluster sizes and different time-warping window sizes.

**Cluster Size**: 128

**Keyword of Interest**: UNIVERSITY (**average length – 0.65 secs)**

**Number of keyword Instance available**: 8

**Time-warping window size**: R = 3

**Utterance being tested: en\_4247\_1\_4.wav (11.47 secs)**

*“resigned here last week effective august 10th and I will be taking the job of the director of English language a.. the English language program.. of a private* ***university*** *a.. in Atlanta”*

**Total segments**: 1758



1 2



3 4



5 6



7 8



9

##### Figure 6.1. Distortion plot for each keyword template (1-8) and the average distortion score from all keyword templates (9). The lowset discortion segment indicated by the green arrow (6.1.9) is the lcoation of the keyword

It can be seen in the final result (Average Distortion Plot) that there is minimum distortion around segment 1500 which can also be seen on 7 out of 8 of the individual keyword scores. Another point worth noting is that not all keyword templates are successful at detecting the keyword in the utterance. The 8th keyword template (Figure 6.1.a) shows no sign of the keyword in the utterance. Keywords from same speaker are easier to spot as results in Figure 6.1.a 2,3 and 4 suggest.

The next experiment is for utterance with no keyword in it. It can be seen that there is no distinct low distortion segments like the one shown in Figure 6.1.

**Utterance being tested: en\_4247\_1\_noword\_2.wav (11.47 secs)**

*“laugh… well I know what it’s like because I.. having lived over in Korea for almost 3 years.. it’s.. you know when you come back it’s all of a sudden.. you know the society has changed a little”*

**Total segments**: 1756



a b



c d



e

##### Figure 6.2. Distortion score plot for each keyword (a-d) and the average for utterance with no keyword (e).

**Utterance being tested: en\_4247\_1\_2.wav (11.47 secs)**

“the English language program.. of a private **university** a.. in Atlanta… a larger **university”**

**Total segments**: 906



##### Figure 6.3. Average Distortion score plot utterance with two keywords.

**Utterance: en\_4184\_1\_2 .wav (12.15secs)**

*“and so he asked… her to have me call him yesterday and it turns out that he was talking to the president of the* ***university*** *of San Diageo... the day before… and told her about..”*

**Total segments**: 1865



##### Figure 6.4. Distortion score for utterance with keyword.

In Figure 6.4, the minimum distortion score occurs at around segment no. 1000 but is not very noticeable.

## Decision Criteria

The average score plot shows a minimum distortion at the location of the keyword. However the minimum distortion is not distinct in all the cases. Moreover, the average score also requires a threshold value in to order to make detection decision. In order to avoid this problem, a voting based algorithm is used as described in Section 5.2.2. The top ten lowest distortion segments are selected for each keyword template and the frequency of occurrence for each segment is counted in different keyword templates. If the segment falls within the top ten scorers for more than half the keywords, it is considered being the keyword.

## System Operating Characteristic

The system can be tuned by varying different parameters. Given below are discussions of those parameters.

### Detection Count

Each keyword template can result in various detection locations in the utterance. The final decision is based on the count of common detected location for available keyword templates. The higher the count, the greater is the confidence of detection. Experimentally, the detection threshold count is set to N/2 where N is the number of keyword templates available. In other words, if half the keyword templates indicate there is keyword present at certain location of the utterance, it must be the keyword.

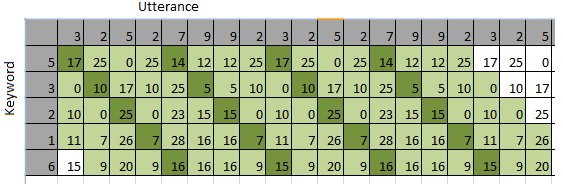
### Segment Span

In segmental time warping comparison, the utterance is divided into overlapping segments and each segment is compared with the keyword of interest. In order to segment the utterance, the width of the segment needs to be defined in advance. A starting point is chosen and a margin of half the segment width is allowed in either side of the point. The actual starting point of the keyword can lie within that width for the particular segment.

Segment width is defined in terms distance between centers of corresponding segments R called the segment span.

Segment Width (Ws) =2\*R + 1

For example, as shown below in Figure 6.5, segment span R = 3 means there is 3 units margin in either side of the center and the segment width is Ws=2\*3+1=7 units. The center of the next segment is R+1 unit from the center of current segment. As a result, there is an overlap of R units between corresponding segments. Each unit is in fact a 20ms window of speech quantized into a codebook index.



Segment -1

Segment -2

Segment -3

##### Figure 6.5. Utterance segmentation for R=3. Segment width Ws=2\*R+1=7.

As seen in Figure 6.5, the un-highlighted portion of the utterance (start margin and end margin) is not included in the computation because it is too short to be a keyword. In this system, the end margin is set to be quarter of the average keyword length. In other words, the last portion of the utterance, equal to the quarter the length of the keyword, is discarded because it is too short to be a keyword.

Segment span can control the system computational speed as well as accuracy. Shown below is the plot of system performance as a function of segment width. Results were obtained by testing on three different keywords out of total fourteen keywords.



a.



b.



c.

##### Figure 6.6. System Performance as a function of segment span – R. (a.) Hits vs R, (b) Misses vs R, and (c) False Alarms vs R.

As seen in Figure 6.6 above, segment span controls the overall system performance and will be the main focus of this discussion. The number of hits increases with the increase in segment span. A segment width of unity (R=0) means that each frame of the utterance segment is forcibly aligned with that of the keyword for comparison. Wider the segments are, more flexible is the comparison allowing more alignment variation. If a keyword is detected in segment N, it means that the keyword is located at N±R – N+KL±R where KL is the keyword length. This means that the precision error of locating the keyword can be up to twice the segment span. Therefore as R increases, the hit count drops due to the decrease in the precision of detection. This also results in increase of misses. False alarms, on the other hand increases with segment width because of flexible comparison.

##### Figure 6.7. Execution time per keyword template per minute of utterance.

Number of segments is inversely proportional to the segment span. The relation is given by:

*segmentCount= (UL-margin-1)/R + 1* Equation 6.1

Where UL is the utterance length, margin is the end margin and R is the segment span.

Keeping in mind all these trade-offs the optimum segment span is in between 5-6.

For a system with 6 keyword templates using a segment margin of 5, the execution time per minute of utterance is around 25 seconds which is less than half the utterance length. This implies that the system can be operated in real time.

## System Evaluation

Using the settings derived from the above discussion, experiments were performed on Call-Home as well as Switchboard corpus. Presented below is a summary of results on some keywords from the Call-Home database.

###### Table 6.1. Detection result for cluster of size 512 and segment span of 5.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Keyword | Templates Available | Hits | False Alarms | Misses | Utterance Length (sec) |
| University | 6 | 8 | 1 | 0 | 111.86 |
| Computer | 4 | 4 | 1 | 1 | 40.09 |
| Language | 4 | 8 | 2 | 3 | 210.53 |
| English | 9 | 12 | 5 | 0 | 343.07 |
| School | 7 | 5 | 2 | 1 | 196.48 |
| Zero | 7 | 3 | 1 | 1 | 96.46 |
| Conversation | 7 | 12 | 2 | 1 | 153.26 |
| College | 7 | 5 | 2 | 1 | 96.73 |
| Program | 6 | 4 | 1 | 2 | 141.77 |
| Relationship | 10 | 8 | 2 | 2 | 119.52 |
| Something | 7 | 5 | 3 | 3 | 484.22 |
| Student | 11 | 8 | 4 | 1 | 81.62 |
| Tomorrow | 10 | 9 | 2 | 1 | 158.35 |
| Funny | 5 | 4 | 2 | 1 | 91.08 |

The accuracy of the system is defined as:

Accuracy = Equation 6.2

Table 6.1 is sorted in order of accuracy and is shown below in Table 6.2.

###### Table 6.2. Results for different keywords sorted in terms of accuracy.

|  |  |  |  |
| --- | --- | --- | --- |
| Keyword | Accuracy | Templates Available | Utterance Length (sec) |
| University | 0.89 | 6 | 111.86 |
| Conversation | 0.8 | 7 | 153.26 |
| Tomorrow | 0.75 | 10 | 158.35 |
| English | 0.71 | 9 | 343.07 |
| Computer | 0.67 | 4 | 40.09 |
| Relationship | 0.67 | 10 | 119.52 |
| School | 0.63 | 7 | 196.48 |
| College | 0.63 | 7 | 96.73 |
| Language | 0.62 | 4 | 210.53 |
| Student | 0.62 | 11 | 81.62 |
| Zero | 0.6 | 7 | 96.46 |
| Program | 0.57 | 6 | 141.77 |
| Funny | 0.57 | 5 | 91.08 |
| Something | 0.45 | 7 | 484.22 |

From Table 6.2, it can be seen that, in general, detection is more accurate for longer keywords. More importantly, here are some parameters on which the results were highly dependent.

* Keyword Segmentation: Template based matching requires one or more keyword templates which had to be isolated from the speech stream. When segmenting the template keyword, one has to be very careful to include entire portion of the keyword and more than that not to include leading or following sounds. If the template keyword has any trace of leading or following word, it can decrease the detection count and at the same time increase chances of false alarm. The bigger challenge is to identify the word boundary accurately. As a matter of fact, it required a number of iterations to get good keyword templates.
* Number of Keyword Templates: Keyword templates are the key to template based detection. But it does not necessarily mean that larger number of templates increases the detection rate. In fact the N/2 voting rule requires more votes for a segment to be considered as a keyword if there are more templates available. Outlandish keyword templates deteriorates detection rate. In , a user based relevance feedback technique after every detection is proposed, which basically assigns different weight to each template after every observation, thereby gradually emphasizing generic templates over odd ones. In the above experiments, 6-7 templates produced the best results.
* Keyword Characteristic: A keyword detection system favors certain keywords over others. From Table 6.2, it can be seen that the system, in general, favors longer keywords over short ones. More than that, the accuracy depends on the use of the keywords. For example words such as UNIVERSITY, RELATIONSHIP or CONVERSATION are pronounced more separated from other words where as words such as SOMETHING are spoken out of context to fill gaps and have larger variation.
* Cluster Size: Experiments were performed on cluster size ranging from 64 to 512. It was found that the detection rate was independent of the cluster size. In general system, larger cluster size would mean more computation especially for the training portion. But due to pre-computation of the distance matrix, once the cluster is trained the execution time does not depend directly on the cluster size.

Accuracy of up to 90% was obtained for some keywords. The precision in determining the location of the keyword was above 70% and precision error of more than 30% was considered a false alarm. The results are comparable to those obtained by using segmental time warping over Gaussian Posteriorgrams [Anguera, et al. 2010, Yaodong Zhang, et al. 2010].

# Conclusion

A keyword spotting system based on segmental time-warping of quantized features was designed and evaluated. First a feature space was created using considerably long and diverse sounds which were then quantized using K-means clustering algorithm into finite clusters. Keyword templates and speech utterance were then converted into 1-dimensional index strings using the codebook. The utterance was then divided into overlapping segments with a certain segment span. Each segment was compared against template keyword(s) in a flexible time scale called the Dynamic Time Warping. Score for each segmented produced the likelihood of the segment being the keyword. A voting based technique was used for detection criteria using N/2 rule, i.e., if half of the keyword templates say that a particular segment was the keyword, it was considered a keyword.

The proposed algorithm does not require transcribed data for training purpose. Once the feature space has been populated, with sufficiently diverse sample utterances that contain enough sound variations, no additional knowledge of underlying language model is required to identify the keyword, thus making the system language independent. It is not just words that can be searched using this algorithm, any phrase or sound can be searched equally efficiently provided the training data consists of those kinds of sounds.

The accuracy of the system is comparable to other Segmental-DTW based system described in [Anguera, et al. 2010], [Yaodong Zhang, et al. 2010]. The major difference in the proposed system, as compared to those systems, is that they are based on Gaussian Posteriorgrams, i.e., the feature space once populated with enough training data is divided into Gaussian Mixtures and posterior probability of every segment belonging to each model is computed. The clustering algorithm used in the system is far simpler as compared to GMMs in terms of memory and computation. The only parameter that is required is the mean of each cluster and likelihood is based on Euclidean distance which is easily computable. Furthermore, the distance can be pre-computed and stored into a distance matrix to optimize the detection process.

Likelihood of a speech frame belonging to a cluster class is given by the Euclidean distance between them which is far more computationally efficient as compared to computing posterior probabilities. Euclidean distance is used as the measure of distance between keyword frames and speech utterance frames which is easier to compute as compared to the log of dot product of two posterior probabilities.

The training process was much simpler and faster compared to training a HMM system or Gaussian Mixture Models. For these experiments, 70 minutes of data from various speakers was used to populate the feature space and the quantization took less than 70 minutes for 512 clusters. The speed of keyword detection depends on the segment span and number of template keywords available. For the experimental setup described above, 7 templates were used in average with a segment span of 5 (R=5). The detection process took half the time of utterance length on average. Algorithms using Gaussian Posteriorgrams, also called PG-DTW [Hazen, et al. 2009], are much more complicated and require much more processing time. As cited in [Yaodong Zhang, et al. 2010], it took them ten minutes to process one hour of speech data in a distributed computing system with 200 CPUs. This data seems impractical for general computing devices such as a personal computer or a mobile device.

Hence, the algorithm is feasible for practical purposes and with further improvements can replace existing complex context independent keyword spotting systems.

# Future Work

From the above mentioned experiments, maximum accuracy of 90% was achieved for some keywords. But it is worth mentioning that the accuracy was low for other keywords. There can be number of possible explanations for it. Some of them are discussed in this section.

The major contribution to low accuracy goes to inaccurate segmentation of template keywords. Especially if the template is coupled with leading or trailing sound from other word, it introduces noise to the system and drops the detection rate. Extraction of keyword templates was the most time consuming task. Therefore the next set of experiments could be to study the effect of incomplete keyword templates and noisy templates on the system. Another way to address this issue would be to implement a user-driven relevance feedback technique as described in . This method would assign different weight to the keyword templates after every iteration thereby increasing the system efficiency.

The performance speed of the system is fast enough for real time application. It can still be optimized by implementing different optimization algorithms as described in [Anguera, et al. 2010]. The method is called Unbounded Dynamic Time Waring (U-DTW) which computes the similarity of two sequences only when needed using a forward-backward path finding algorithm which brings significant computational savings [Anguera, et al. 2010]. The basic idea is proper selection of the synchronization points (SPs) which are used as the starting point to look for possible matching segments. The computation time was shown to drop from 82.7 ms to 10.6 ms per word using this algorithm.

Results obtained from experiments were positive and promising. But additional results are required to analyze the system sensitivity and robustness to noise. System evaluation using ROC/DET curve could not be completed due to limitation of time and data. The system was operated at optimal point as suggested by some preliminary experiments which could be more convincing if supported by more testing data under different parametric conditions. System evaluation based on length and composition of keyword could be done more thoroughly. In other words the system requires much more training and detailed evaluation.

# References

1. DONGZHI HE, YIBIN HOU, YUANYUAN LI AND ZHI-HAO DING. 2010. Key technologies of pre-processing and post-processing methods for embedded automatic speech recognition systems. In *Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference*, 76-80.
2. ZAHARIA, T., SEGARCEANU, S., COTESCU, M. AND SPATARU, A., 2010. Quantized Dynamic Time Warping (DTW) algorithm. *Communications (COMM), 2010 8th International Conference*, 91-94.
3. GRANGIER, D. AND BENGIO, S., 2007. Learning the Inter-frame Distance for Discriminative Template-based Keyword. International Conference on Speech Communication and Technology (INTERSPEECH).
4. SAKOE, H. AND CHIBA, S., 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing,* 26, 43-49.
5. MYERS, C., RABINER, L. AND ROSENBERG, A., 1980. An investigation of the use of dynamic time warping for word spotting and connected speech recognition. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '80*, 173-177.
6. ANGUERA, X., MACRAE, R. AND OLIVER, N., 2010. Partial sequence matching using an Unbounded Dynamic Time Warping algorithm. *IEEE International Conference on* *Acoustics Speech and Signal Processing (ICASSP), 2010*, 3582-3585.
7. PALIWAL, K. AND WOJCICKI, K., 2008. Effect of Analysis Window Duration on Speech Intelligibility. *Signal Processing Letters, IEEE* 15, 785-788.
8. IYER, A., OFOEGBU, U., YANTORNO, R.E. AND SMOLENKI, B., 2009. Speaker distinguishing distances: a comparitive study. International Journal Of Speech Technology 2-3, 95-107.
9. GOPALAN, K., CHU, T. AND MIAO, X., 2009. An Utterance Recognition Technique for Keyword Spotting by Fusion of Bark Energy and MFCC.  International conference on signal, speech and image processing.
10. ZWICKER, E., 1961. Subdivision of the Audible Frequency Range into Critical Bands(Frequenzgruppen). *The Journal of the Acoustical Society of America* 33, 248-248.
11. HERMANSKY, H., MORGAN, N., BAYYA, A. AND KOHN, P., 1992. RASTA-PLP speech analysis technique. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference*, 121-124.
12. PARK, A.S. AND GLASS, J.R., 2008. Unsupervised Pattern Discovery in Speech. *IEEE Transactions on Audio, Speech, and Language Processing,* 16, 186-197.
13. ZHANG, Y. AND GLASS, J.R., 2009. Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams. *IEEE Workshop on Automatic Speech Recognition & Understanding, 2009. ASRU 2009,* 398-403.
14. HAZEN, T.J., SHEN, W. AND WHITE, C., 2009. Query-by-example spoken term detection using phonetic posteriorgram templates. *IEEE Workshop on Automatic Speech Recognition & Understanding, 2009. ASRU 2009*, 421-426.
15. KESHET, J., GRANGIER, D. AND BENGIO, S., 2009. Discriminative keyword spotting. *Speech Communication* 51, 317-29.
16. SOMERVUO, P. AND KOHONEN, T., 1999. Self-organizing maps and learning vector quantization for feature sequences. *Neural Processing Letters* 10, 151-159.
17. MORGAN, D.P., SCOFIELD, C.L., LORENZO, T.M., REAL, E.C. AND LOCONTO, D.P., 1990. A keyword spotter which incorporates neural networks for secondary processing. *International Conference on* *Acoustics, Speech, and Signal Processing, ICASSP-1990., 1990,* 113-116 vol.1.
18. BRONKHORST, A.W., 2000. The cocktail party phenomenon: A review of research on speech intelligibility in multiple-talker conditions. *Acta Acustica united with Acustica* 86, 117–128.
19. SILAGHI, M.C., 2008. A new evaluation criteria for keyword spotting techniques and a new algorithm. Proceedings of InterSpeech, 2005.
20. LLOYD, S., 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory,* 28, 129-137.
21. ZHANG, Y. AND GLASS, J.R., 2010. Towards multi-speaker unsupervised speech pattern discovery. *IEEE International Conference on* *Acoustics Speech and Signal Processing (ICASSP), 2010,* 4366-4369.