DESIGN OF AN AUTOMATIC WORD BOUNDARY DETECTION SYSTEM USING THE COUNTING RULE

A Thesis
Submitted to
the Temple University Graduate Board

In Partial Fulfillment
Of the Requirements for the Degree
MASTER OF SCIENCE
in Electrical Engineering

By
Sandeep Kanneganti
January, 2011

Dr. Robert Yantorno
Thesis Advisor
Electrical and Computer Engineering

Dr. Joseph Picone
Professor and Chair
Electrical and Computer Engineering
Committee Member

Dr. Dennis Silage
Professor
Electrical and Computer Engineering
Committee Member

ABSTRACT

Word boundary detection is the stepping stone for many applications like keyword spotting, speech recognition, etc. It is proved that fifty percent of the speech recognition errors are due to the errors in the word boundary detector. Efficient word boundary detection can reduce the recognition errors and improve the performance of keyword spotting algorithms. Word boundary detection also helps in reducing the search space in the keyword spotting algorithm. Speech is non-stationary in nature and most of the time no utterance of the same word will be same as another utterance of same word. This makes it challenging to develop any speech processing algorithm.

Many algorithms, to detect word boundaries, use acoustic features, lexical cues, energy, pitch etc. Acoustic features of energy, pitch and Teager Energy were used in this research to detect word boundaries. The strengths and drawbacks of each of the techniques are identified and the information from all the techniques was fused to obtain improved word boundary detection. Energy was able to detect word boundaries with 56% of the time, pitch with 68% of the time and Teager Energy with 72% of the time

Simple counting rule, which is based on reinforcement learning, was used in this research to fuse the detections from the three techniques to make a final decision on the word boundaries. This system does not need prior knowledge about the detection and false alarm probabilities of the techniques. The weights are adapted with the outcome in every iteration. Fusion of the decisions from energy, Teager Energy and pitch yielded a 79% hit rate on spontaneous speech using counting rule whereas linear opinion pool and log opinion pool produced 73% and 74% hit rate respectively.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Word Boundary Detection

Word Boundary Detection (WBD) is defined as identifying the start and the end of each word in a spoken utterance. Detecting the word boundary is used in many applications like keyword spotting, speech recognition system etc. In this research various types of word boundary detection environments which include noise free, telephonic conversations are considered.

Two types of speech are considered in this research which are explained below:

- Constrained Speech: In the constrained speech, the utterance and words has well defined boundaries with well defined pauses between words.
- Unconstrained Speech: Speech with no well defined boundaries, grammar or pauses is called unconstrained speech. Word boundary detection is challenging in the unconstrained speech and without a good detection algorithm word boundary detection can result in the many false alarms and misses.

Word boundary detection can also be classified as speaker dependent and speaker independent. Speaker dependent systems are easy to develop when compared to speaker independent systems as one can have the information about utterance characteristics of the single speaker like pitch, but speaker independent systems have to be developed based on many speakers which makes it challenging due to different prosody, pitch etc. In this research a speaker independent system is developed.

1.2 Problem Statement

The goal of the research presented here is to detect the word boundaries in a spontaneous unconstrained speech independent of the speaker.

1.3 Previous Research

Word boundary detection is usually performed using the acoustic information of speech (Janqua [1994], Wang [2003] and Cettolo [1998]). Janqua *et al* [1994] used energy based on sub-band analysis to detect the clues for word boundary detection even in the presence of noise. Using this algorithm the recognition errors were reduced to 20% which was 50%. Wang *et al* [2003] presented sentence boundary detection using pause, rate of speech and prosody. In the first stage basic features frame energy, Zero Crossing Rate (ZCR) and pitch are calculated. In the second stage pauses, rate of speech and prosodic features are obtained to mark candidate word boundaries. In the third stage a statistical method Adaboost is used to mark the sentence boundaries based on the contextual information. The hit rate using this method is 79%. Andreas Tsiartas *et al* [2009] proposed using lexical features along with acoustic features which greatly improved the word boundary detection rate. Acoustic features like short time energy, short time pitch frequency and short time zero crossing rate are used along with the Boundary Confidence Coefficient (BCC) parameter which is based on lexical information of the frame. These lexical and acoustic features are used to train the Hidden Markov Model (HMM) based classifier to detect word boundaries. The word boundaries were detected with F-score of 0.81. Some word boundary detection techniques are discussed in this section

1.3.1 Robust Word Boundary Detection in Spontaneous Speech Using Acoustic and Lexical Cues

Acoustic cues often fail to give cues about the word boundaries when the beginning of a word gets co-articulated with the end of the previous word, which is common in spontaneous speech. Tsiartas *et al* [2009] used lexical cues along with acoustic features to detect word boundaries. Although lexical cues can be erroneous, they provide rough and potentially imperfect word segmentation information which when used with additional acoustic information can produce improved word boundary detection.

The acoustic features used by Tsiartas *et al* [2009] are:

- Short time energy: The signal energy variation can give a cue to determine if the frame is close to word boundary.

- Short time zero crossing rate: If there is a reduced speech activity from a word to word boundary, it can act as a cue to detect the boundary.
- Short time pitch frequency: Pitch can be used to detect the unvoiced sound near a word boundary which can act as a cue.

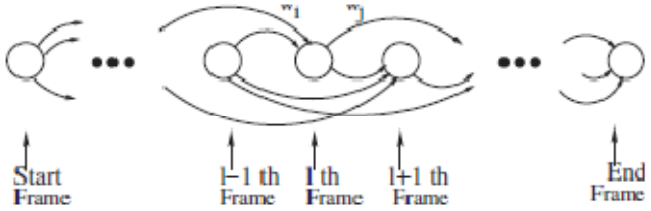

Figure 1.1: A typical word Lattice in Automatic Speech Recognizer

The authors considered the word lattice at the each frame where the nodes (the circles) in the **Figure 1.1** indicate possible word boundary and arcs show different possible words. Using a language model, the most probable paths are listed in the order of their probability in the entire lattice. A new parameter Boundary Confidence Coefficient (BCC) at frame **i** is defined as:

$$\text{BCC(i)} = \frac{1}{N}\sum_{k=1}^{N} g(p\,(k)) \tag{1.1}$$

$$\text{g(p(k))} = \begin{cases} 1 \; if \; p(k) > th \\ 0 \; otherwise \end{cases} \tag{1.2}$$

Where **p(k)** is the probability of k^{th} path in the lattice and ***th*** is the optimally chosen threshold of the probability. The authors used 5000 utterances from SwitchBoard for training purpose and 2000 for the testing purpose. The features described above were used to train a Hidden Markov Model (HMM) based classifier and trained the detector using only clean speech. Two symbols, a boundary symbol and a non boundary symbol, are used in the HMM. In evaluating this model the authors used speech with different types of noise and different SNR levels between 5 and 20 dB. It was observed that use of the lexical features along with acoustic features always produced higher detection rate when using only the acoustic features. As the SNR decreased BCC feature also suffered which reduces the boundary detection. The results are shown on F-score basis which is measure of tests accuracy.

$$\text{F-score} = 2^{*}\,(p\,.\,r)/\,(p+r) \tag{1.3}$$

Where **p** = number of hits/ total boundaries detected

r = number of hits/ number of actual word boundaries

The word boundary detection F-score is 0.81 using both lexical and acoustic features is much higher than using only the acoustic features which has a F-score of 0.63

1.3.2 Word Boundary Detection with Mel-Scale Frequency Bank

An Adaptive Time-Frequency (ATF) parameter for extracting both the time and frequency features of noisy speech signals for word boundaries was proposed by Der Wu *et al* [2000]. Mel scale frequency bank is approximated by simulating 20 triangular band pass filters f(i,k) ($1 \le i \le 20$, $0 \le k \le 63$) over a range of 0 – 4000 Hz. Each filter band has a triangular bandpass frequency response and the spacing is determined by a constant mel frequency interval given by the equation below:

$$\text{Mel} = 2595 \log(1 + f/700) \tag{1.4}$$

Energy of each frequency band can be calculated for each time frame of the signal using the formula

$$F(m) = x(m,i) = \sum x_{freq}(m,k)\, f(i,k)\ ,\ 0 \le m \le M-1,\ 0 \le k \le N-1,\ 0 \le i \le 20 \tag{1.5}$$

where **i** is filter bank index, **N** is N point DFT, **k** is spectrum index, **m** is frame number, **M** is number of frames for analysis, $\mathbf{x_{freq}}$ is DFT of $x_{time}(m,n)$, **f(i,k)** weighting factor of frequency energy.

$$\text{Time energy } T(m) = x_{rms}(m) - \sum_{m=0}^{4} x(m)/5 \tag{1.6}$$

$$\text{ATF}(m) = \text{Smoothing}(T(m) + cF(m)) \tag{1.7}$$

Smoothing is performed by a three point median filter, **c** is a proper weighting factor and **T(m)** is time energy, **F(m)** is frequency energy. The algorithm first performs a noise classification procedure to determine noise level (high, medium or low) and the noise category (high or low zero-crossing rate) by

using ten frames of relative silence at the beginning of the recording and computing an average of the logarithm of the rms energy and the zero-crossing rate for these frames. It then uses the ATF parameter to find the islands of reliability boundary using a threshold. It adjusts the word boundary from 50 ms to 150 ms based on the noise level. Using the ATF parameter reduced the recognition error due to end point detection to about 20 % which was 30 % using only time frequency parameter. It is shown that the ATF parameter can extract useful frequency information of the signal by using the maximum total energy bands without the need to determine the SNR and recognize the types of noise.

1.3.3 A Robust Word Boundary Detection Algorithm

Accurate detection of word boundary is very important in the performance of speech recognizers. An evaluation of speech recognizer showed that more than half of the recognition errors were due to word boundary detector (Agaiby [1997]). Agaiby *et al* [1997] showed a word boundary detection algorithm that performs well under a variety of noise. The objective of the algorithm presented by Agaiby [1997] was to maintain computational simplicity while detecting word boundaries efficiently even in a noisy environment. The algorithm uses direction of the signal, using two microphones, as the main criteria to distinguish between wanted speech and background noise. Direction is the simple observation that in most of the speech recognition applications the position of the speaker can be considered to be within a predefined area facing the microphones. It is mentioned that it is easier to adjust the position of the user than to control the types or levels of background noise. The algorithm is based on the use of two microphones, and is therefore able to estimate the direction of the signal source and not its position. It uses the time delay between the received signals at the two microphones which are 20 cm apart to estimate the dominant signal source direction. This estimate along with an estimate of coherence function between two signals are used to determine the initial values for word boundaries and then are refined using the signal energy by adding the head and tail frames at the beginning and the end of the wanted speech.

The noisy speech is first processed using manually labeled speech and the recognizer is evaluated and then the word boundaries are detected using the algorithm discussed and the recognizer is evaluated. A significant improvement of 45% was achieved in the recognition accuracy with different SNR levels. This

algorithm is capable of differentiating between desired speech and other types of noise including competing real time talkers can be implemented.

1.3.4 Detection of Prosodic Word Boundaries

A reliable method of prosodic word boundary detection for continuous Japanese speech based on the statistical modeling of fundamental frequency contours of prosodic words was developed by Hirose *et al* [2000]. Prosodic word boundary is defined as a word or a word chunk corresponding to an accent component which is otherwise called accent phrase. Taking into account “Mora” (the basic unit in Japanese like syllable for English) and that its relative fundamental frequency (F_0) value is important for accent-type perception Hirose [2000] developed a modeling scheme. The models are time-aligned to segmental boundaries they can be easily incorporated into phoneme based recognition. The modeling in Mora units has an advantage over frame based as it can be robust to F_0 contour fluctuations at consonants and can be trained using a small sized corpus.

A sentence fundamental frequency contour in logarithmic frequency scale is segmented into mora units using mora boundary information obtained by the phoneme recognition process. The segmented F_0 contours are represented by pair of codes, one representing the contour shape and other representing the average shift from preceding Mora. Based on the accent types and the succeeding pauses the prosodic words are modeled. These models are matched against the input utterances to obtain the prosodic word sequence with their accent types and prosodic word boundaries detected simultaneously. Using this modeling the detection rate was 75%. The authors also stated that a corpus with prosodic boundaries can greatly improve the recognition rate.

1.3.5 A Multi-pass Linear Fold Algorithm for Sentence Boundary Detection

Detecting sentence boundary and using the cues of the sentence boundary can improve the word boundary detection rates. A multi-pass linear fold algorithm for sentence boundary detection in spontaneous speech was proposed by Wang *et al* [2004]. It uses only prosodic cues and does not rely on

segmentation information from a speech decoder. Pause and pitch features were found to play a crucial role in boundary and disfluency detection problems. By analyzing a number of files from the SwitchBoard corpus it is observed that sentence boundaries show larger break duration and the pauses are large when compared to other pauses like intra and inter word breaks. In this algorithm Wang [2004] mainly focused on pitch breaks, pitch properties in the neighborhood of the pauses. After the pitch detection, the pitch break durations are plotted in ascending order of the pitch breaks but not in the way they occur in the utterance. After plotting the pitch breaks, it is observed that the region corresponding to shorter duration has most of the pitch break points and is related to intra and inter word breaks. The region with larger duration is sparsely populated. Points in the sorted pitch break map are cut into two groups and a linear fit is done to each group independently. The cut which has minimal mean squared deviation is chosen as shown in the **Figure 1.2**.

As we can see in the **Figure 1.2** this one pass linear algorithm was able to fit the data well most of the time. The upper line in the figure represents the fluent pause, hesitation and other special breaks which can be the sentence boundary. There can be other cases where there is a rich transition with no good pauses between the sentences. In such cases the algorithm is applied recursively removing the lower half of the data in each iteration. After obtaining the results with this multi-pass algorithm the output is post processed on the following criteria:

1) If a pitch break point occurs early in the sentence it cannot be a sentence boundary.
2) Sentence boundaries cannot appear too close to one another.
3) Occurrence of pitch reset is associated with sentence boundary. If the pitch reset was observed it marked as sentence boundary.
4) If there is no other clue available the break point with longest duration is selected as sentence boundary.

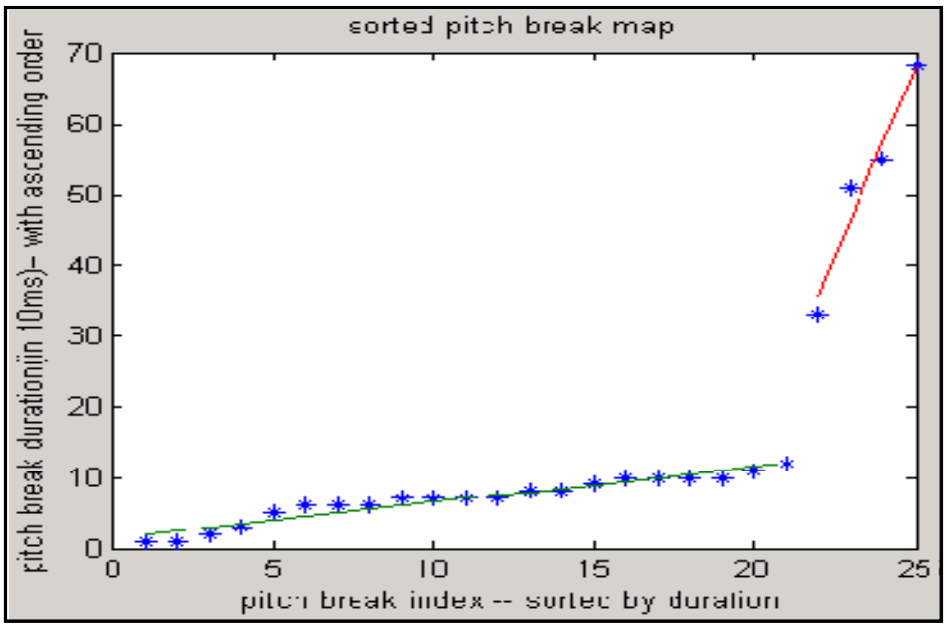

Figure 1.2: Linear Fold Algorithm – Wang [2004]

Based on the above four assumptions the results from the algorithm are modified. Using this algorithm on 100 randomly chosen utterances from Switchboard the overall hit rate in sentence boundary detection was observed to be 75 % where as the best results obtained using range features and a decision tree based was 78%.

1.4 Scope of Research

The aim of this research is to develop an algorithm with an improved word boundary detection rate in common speech signal environments like telephonic conversations. In this research the algorithm is developed for English and it eventually could be modified to suit any stress bound language.

CHAPTER 2

INTRODUCTION TO WORD BOUNDARY DETECTION

Words are the building blocks of speech. Different languages have different ways to start and end words. One of the types of language models is stress bound language, having stress at the beginning of the words, which includes English. Techniques like pitch, energy and Teager Energy take advantage of stress in the stress bound languages and can be efficient in identifying the beginning of the word boundary. For word boundary detection in English language, three features, having the acoustic information were used. Each of the techniques is used to detect the word boundaries individually, to mark the candidate word boundaries and then the results from all the techniques will be fused together to identify the word boundaries. The techniques are stated below

1) Segmentation to Voiced/Unvoiced/Silence
2) Energy
3) Teager Energy
4) Pitch

2.1 Segmentation

The knowledge of acoustic features in particular voiced and unvoiced segment plays an important role in many speech analysis systems. Segmentation of speech into voiced, unvoiced and silence could help in marking the candidate word boundaries in the speech. Voiced or Unvoiced speech followed by silence could be a candidate word boundary and in the similar way, silence followed by the voiced or unvoiced speech could be the candidate word boundary. So the segmentation of speech into voiced/unvoiced/silence is important in word boundary detection. Based the characteristics of the voiced and unvoiced speech listed in **Table 2.1**, many algorithms were developed to detect voiced and unvoiced speech Arifianto [2007], Brett's segmentation algorithm.

One algorithm developed for segmentation of speech was by Arifianto *et al* [2007]. He developed a new measure called the Instantaneous Frequency Amplitude Spectrum (IFAS) which is a measure of harmonicity of a speech segment, which is better than the short time Fourier transform (STFT). Voicing decision threshold relies on the difference of the harmonicity measured by the IFAS. It is quoted that major problem occurs at the segment where transition occurs from voiced to unvoiced or vice versa.

Table 2.1: Differences in Voiced and Unvoiced speech

Voiced Speech	Unvoiced Speech
Energy per frame is much higher	Energy per frame is less
Less zero crossings	More Zero crossings
Quasi periodic	Noise like

To reduce the error rate in the transition regions Arifianto [2007] the used two parameters namely entropy of the magnitude spectrum and instantaneous power of the signal. After the estimation of all the parameters above, the voicing decision is based on two techniques, direct threshodling and Peak-Picking. Using these techniques the results obtained were good with very low error rate of as much as 13.9 percent.

Problems faced in segmentation:

1) Unvoiced and voiced speech overlap with each other making it difficult to identify their exact end points.
2) Low energy voiced speech, like whisper, can be easily be mistaken as unvoiced speech.

2.2 Energy

Energy is one of the basic acoustic features in speech. It is mainly used in the Voice Activity Detection (VAD). If one can detect the voice activity it is easy to mark the word boundaries using other techniques. Li *et al* [2008] used the speech energy edge information which is the change in energy of the incoming speech. It has a capability of detecting voicing activity in low SNR environments. There are two steps, the

first step is to employ an optimized edge filter to identify the sudden rise or fall in the energy is defined in the equation below

$$f(x) = e^{Ax}[K_1 \text{ Sin}(Ax) + K_2 \cos(Ax)] + e^{-Ax}[K_3 \sin(Ax) + K_4 \cos(Ax)] + K_5 + K_6 e^{\pi} \quad (2.1)$$

$$\text{Edge}(g) = \sum_{i=-w}^{w} f(i)g(t+i) \quad (2.2)$$

Where **A** and $\mathbf{K_i}$ are parameters of the edge filter, **s** is a positive constant, **g** is the speech energy, **t** is the current frame number and **W** is the window size. This filter is used to compute speech energy and obtain the edge information. In the next stage a four state finite state is applied with silence, starting speech, during speech and ending speech as its four states. Either silence or starting speech can be the starting point of the incoming speech data and then each frame will be classified into one of the four states of the transitions state diagram as shown as shown in **Figure 2.1** below.

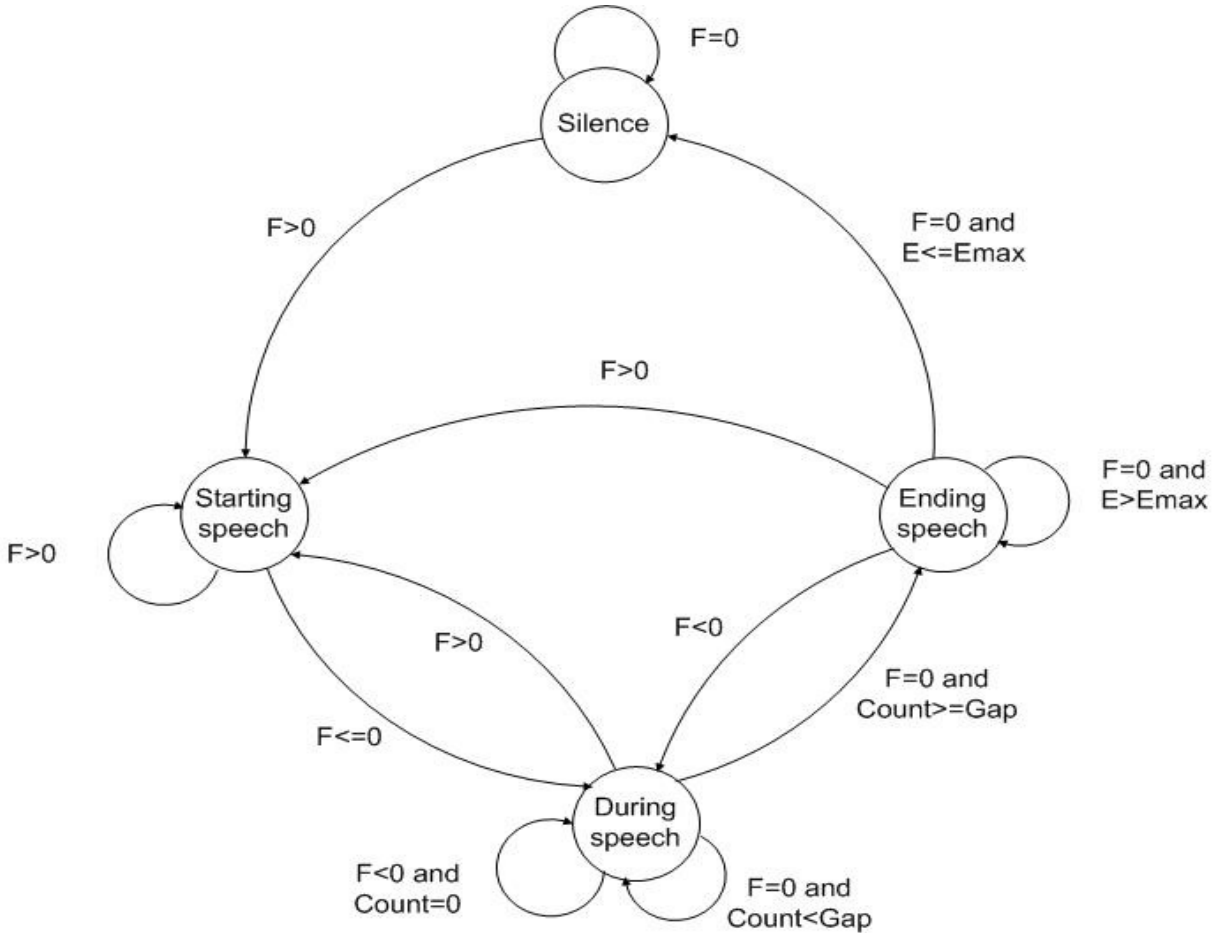

Figure 2.1: Four state finite state diagram

Where **Gap** is the minimum frame number between the detected end point, $\mathbf{E_{max}}$ is the average energy value of the adjacent frames, **F** is the feature value of the current frame. The starting and ending boundaries of the speech are detected using this algorithm, which also identifies the background noise and silence. Using the proposed algorithm the highest recognition accuracy obtained is 80 % using Canadian English.

Xufang Zhao *et al* [2008] developed an automatic speech signal segmentation approach using the silence detection which in turn uses short time frame energy, convex hull energy analysis and spectral variation analysis for the Mandarin language. The convex hull is the hull that exhibits minimal magnitude, monotonically non-decreasing until the loudness peak and monotonically non-increasing. The convex hull algorithm, which evaluates the shape of the loudness pattern, was used to find syllable boundaries. In Mandarin, the Zero Crossing Rate (ZCR) provides adequate spectral information, therefore an onset of a high ZCR is the beginning of a consonant and an offset means the end point of the consonant. A ZCR of 2500 per second is considered as the threshold in this algorithm. Unlike HMM's Zhao's [2008] techniques does not need a language model and is also independent of the accent of the utterance. Initially pauses were detected in the speech segment using short time frame energy and the candidate word boundaries are marked based on a threshold. A hamming window is used twice, first to smooth the short time energy curve in the time domain and second to magnify the valleys of the curve to mark the boundaries. Later the zero crossing rate is used determine the boundaries from the candidate word boundaries. If the zero crossing rate is higher than the threshold, it is identified as unvoiced speech and in mandarin most of the syllables start with unvoiced speech, so it marked as a boundary. Using these techniques they obtained very good results of about 93% hit rate.

Energy can be an important parameter in identifying the boundaries, and voicing activity. However, for automatic speech segmentation, it is difficult to segment the speech reliably on energy changes as these cues often are unreliable due to co-articulation. However it is an important acoustic feature and will produce good results when combined with other techniques like pitch detection in identifying the boundaries.

2.3 Teager Energy

In signal processing, energy in a signal is usually referred to the average of the sum of the squares of the magnitude. Another way to measure energy is to take the discrete Fourier transform (DFT) of a signal segment where the energy is represented by its frequency components. Kaiser *et al* [1990] proved, using basic laws of physics, that energy is proportional to square of amplitude and frequency using simple harmonic motion. Based on that observation the author developed a new parameter called Teager Energy (TE) that is dependent on square of amplitude and frequency. In other words, Teager Energy estimates the true energy of the source of resonance. A formula for Teager Energy is stated below.

$$TE(n) = x(n)^2 - x(n-1) * x(n+1) \quad (2.3)$$

Where **n** is the index of the sample and **x** is the speech signal.

This algorithm operates only on three sequential samples at a time. The Teager Energy operator is:

- Independent of the initial phase.
- Symmetric in time.
- Capable of responding to changes in both amplitude and frequency very rapidly.

The third feature of the Teager Energy operator is of utmost importance for those in speech processing, because speech is non-stationary and the acoustic features change very rapidly. The Teager Energy operator is capable of identifying these changes as it is dependent on only three consecutive samples and also proves to be very efficient. It has been shown that signals with rapidly changing frequency, that the instantaneous frequency is estimated very quickly. Teager Energy has the capability of capturing the energy fluctuations in a glottal cycle.

It was shown by Kaiser [1990] that the expected value of the output of the Teager Energy algorithm is offset by the variance of the noise when compared with clean signal, which shows that the algorithm is sensitive to the noise. The effects of the noise can be reduced by proper linear filtering using a median or moving average filter.

Teager Energy and Mel scale proved to be prominent algorithms in speech processing. Nehe *et al* [2009] introduced a word recognition method using mel frequency and Teager Energy features in a noisy environment. First the short time spectrum of speech is calculated. Next the frame spectrum is passed through a Mel scaled triangular filter bank. Teager Energy operator is applied and then the mean of the absolute values of the sequence is obtained. Finally the cepstral coefficients are extracted by applying the discrete cosine transformation on the mean's obtained. The effectiveness of the Teager Energy operator is illustrated in its ability to recognize the isolated words in the TI-20 word database even in the presence of different types of noises. Teager Energy has many applications in speech processing like word boundary detection, isolated word recognition, pitch detection, audio de-noising, voice activity detection, etc.

2.4 Pitch

Pitch detection is fundamental to many digital speech processing applications like speaker identification and speech recognition. Although there are many pitch detection algorithms based on spectrum, cepstrum, neural networks, etc., very few techniques like short time Average Magnitude Difference Function (AMDF) and short term Auto Correlation Function (ACF) are employed in real time pitch detector according to Li Hui *et al* [2006]. AMDF is computed by taking the absolute magnitude by forming a difference signal between the delayed speech and original speech, at each delay value. AMDF is computationally easy to implement.

$$\text{AMDF(t)} = 1/N \sum_{n=0}^{N-1-m} |x(n) - x(n+t)| \tag{2.4}$$

Where **N** is the frame length, **n** is the frame number and **x** is the speech signal. The major drawback of the AMDF approach, for pitch detection, is that it is highly influenced by the intensity variation of speech in each frame and background noise of the speech signal. In order to overcome these drawbacks ACF is proposed to be used along with AMDF by Xiaokun Li [2008]. First the AMDF values are computed for each frame of the signal. The AMDF values are converted to one bit signals to reduce the computational complexity in ACF and also to decrease the effects of amplitude and formants on pitch detection. The

authors proved that AMDF used along with the ACF had an error of 8% to detect pitch, where as using only AMDF had an error of 15%.

Cepstrum is the inverse Fourier transform of the logarithmic spectrum of the signal. Cepstrum of voiced speech contains strong peaks corresponding to pitch periods. Verteletskaya *et al* [2009] used AMDF and ACF for pitch detection along with Cepstrum Pitch Determination (CPD) and Spectro-Temporal Auto-Correlation (STA) function. For pitch determination, the real part of the cepstrum is sufficient. The cepstral peak, corresponding to pitch, is very sharp, so to determine the pitch using the cepstrum one has to search for peaks in the typical fundamental frequencies (50-500). Temporal auto-correlation is widely used for pitch detection due to its simplicity but it can result in detecting unwanted pitch period multiples while spectral auto-correlation based pitch detection can result in pitch halving. So Verteletskaya [2009] has developed STA method to determine pitch effectively. A parameter Gross Error (GE) is the percentage of voiced speech segments for which the detected pitch is more than 20% higher than the reference pitch. Using the STA method, GE for male is 0.53% and for female is 2.84%.

2.5 Fusion

After obtaining the acoustic information using all the three techniques energy, Teager Energy, pitch, fusion of the results can improve the detection of word boundaries. Some fusion techniques discussed by Punska [1999] are explained in this section.

2.5.1 Bayesian Fusion:

Bayesian posterior probability reflects the belief in the hypothesis based on the current observations and prior information. Let E be an event to be evaluated and x_1, x_2 are the two pieces of information from two different sensors, from Bayes theorem we compute the posterior probability as:

$$p(E|x_1,x_2) = p(x_2|E,x_1)\, p(E|\, x_1)/\, p(x_2,x_1) \qquad (2.5)$$

2.5.2 Linear Opinion Pool

When information originates from different sources, it is important to know how reliable and relevant information is from the sources. To identify the reliability of information from the sources, each source is assigned a weight 'w' and the posterior probabilities from each source are combined linearly to make a decision on the event E.

$$p(E|x_1,x_2,..,x_n) = \sum_{j=1}^{j=n} w_j\, p(E|x_j) \quad (2.6)$$

Where $\mathbf{x_j}$ is the information from $\mathbf{i^{th}}$ source and $\mathbf{w_j}$ is the weight assigned to the source. This method is used to develop the system which models the reliability and weight out the faulty sensors.

2.5.3 Independent Opinion Pool

In independent opinion pool, it is assumed that information from the different sources is independent and the posterior probability is computed as:

$$p(E|x_1,x_2,..,x_n) = \prod_{j=1}^{j=n} w_j\, p(E|x_j) \quad (2.7)$$

This is a difficult condition as the information from all the sensors cannot always be independent and when applying this method care has to be taken to make sure the information from different sensors is complementary.

2.5.4 Independent Likelihood Pool

Independent likelihood pool is better suited if all the information sources have prior information. The Independent Likelihood Pool more accurately describes the situation in multi-sensor systems where the conditional distribution of the observation can be shown to be independent. According to the Bayes theorem, the independent likelihood pool is defined as:

$$p(E|x_1,x_2,..,x_n) = p(E) \prod_{j=1}^{j=n} w_j\, p(E|x_j) \quad (2.8)$$

However, if there are dependencies between the information sources linear opinion pool is the best option.

2.5.5 Logarithmic Opinion Pool (Log-OP)

Logarithmic opinion pool is given by the following equation:

$$p(E|x_1,x_2,..,x_n) = p(E) \prod_{j=1}^{j=n} p(E|x_j)^{w_j} \quad (2.9)$$

Taking log of the product on both sides results in Equation 2.10:

$$p(E|x_1,x_2,..,x_n) = \sum_{j=1}^{j=n} w_j \log(p(E|x_j)) \quad (2.10)$$

Accordingly, the linear opinion pool method can more generally be defined as a sum rule and the logarithmic opinion pool consensus rule can more generally defined as the product rule (Kittler *et al*, 1998).

2.5.6 Simple Counting Rule:

Simple counting rule for optimal data fusion, developed by Mansouri and Fathi [2003], is used to fuse the decisions from the three techniques. This adaptive fusion model estimates the probability of detection (P_d) and probability of false alarm (P_f) based on the counting rule. Some of the advantages of this fusion method are:

- This system does not need prior knowledge about the detection and false alarm probabilities.
- The weights are adapted with the outcome at every iteration.
- The results prove to be effective and the decision error can be reduced.
- Computationally simple.

The counting rule was used to integrate multiple individual structural damage detection measures by Chen *et al* [2010] to form a new measure, and the new measure has higher probability of correct detection than any individual measure. In accordance with the simple counting rule, relative frequency which is the ratio of the numbers of the events is utilized to evaluate the weight factors. This fusion model replaced Maximum Joint Probability (MJP) decision rule as it does not need any prior information. Independently identically distributed detectors with zero mean additive Gaussian random process were used to test the performance of counting rule. Having selected the random noise process, the theoretical probabilities of detection and false alarm for each detector can be readily evaluated. For

the Gaussian case, they can be determined by the standard deviation. The theoretical probabilities of detections and false alarms were calculated to compare with the weights estimated by the counting rule. The weights of each of detector estimated by counting rule are close to the theoretical weights as shown in the **Table 2.2**. The steady state value of the weights is reached approximately after 1000 iterations.

Table 2.2 Comparison Between Theoretical and Steady State Values of Weights

Sensor	1st	2nd	3rd	4th
Theoretical Weight	2.4895	2.4895	1.8721	3.7579
Steady state weight	2.4853	2.4916	1.8721	3.7941

Disaster relief networks are designed to be adaptable and resilient to encompass the demands of the emergency service. Cognitive Radio enhanced ad-hoc architecture has been put forward as a candidate to enable such networks. Nuno *et al* [2009] proposed a cluster based orchestration cooperative sensing scheme, which adapts to the cluster nodes surrounding radio environment state as well as to the degree of correlation observed between those nodes using counting rule. This system consists of a number of local detectors and a fusion center, where the local detectors make a decision of the underlying binary hypothesis testing problem based on their own observations and then transmit their decisions to the fusion center where the global decision is derived using counting rule. It is shown that Root Mean Square Error (RMSE) using the counting rule is 20% which outperformed AND rule with an RMSE of 25%.

CHAPTER 3

RESULTS

In this chapter, results from the experiments using three different features, energy, Teager Energy (TE), and pitch, to detect the word boundaries are discussed. The detections from the three techniques are then fused using simple counting rule [Mansouri and Fathi [2003]] to make a decision on the word boundaries and is compared with linear and log opinion pool.

3.1 Database:

The database used in this research is Switchboard, which has recorded telephonic conversations of spontaneous speech. Utterances from thirty randomly selected speakers which include 17 male and 13 female speakers were used. Around three thousand word boundaries were used to test the performance of the algorithm. Each speech signal was sampled at 8 KHz and the length of the utterances used in this research varied between 60 seconds to 180 seconds. The actual word boundaries used to compare the results were obtained from the transcriptions of the Switchboard database.

3.2 Short Words:

Since the speech used is spontaneous, many boundaries are not well defined and often many words are not separated by pause or silence. This problem is most noticeable when really short words like no, ok, or, I, were encountered in speech. In the Switchboard database, the percentage of short words, which are less than 0.1 second of word length, was determined. Twenty four thousand words were used to estimate the percentage of short words and there were approximately 12 percent of short words or 2817 short words. The word boundaries of these words were missed, except when they are separated by pause with their neighboring word, in most of the cases. Word boundaries of long words were detected with higher hit rate than boundary detection of short words. The 12% of short words contribute to majority of misses in this algorithm. It is not possible to avoid the word boundary detection of short words. Therefore the length of short words which is 0.1 second or 800 samples (or less) is used as a threshold to avoid identifying too close candidate word boundaries discussed in the next section.

3.3 Avoiding Close Candidate Word Boundaries

From previous section it was observed that 88% of the words are of length 0.1 seconds or more. It was observed that in each of the three techniques, energy, Teager Energy, pitch and fusion, many candidate word boundaries were marked with distance between them much less than the length of short words which resulted in many false alarms. To avoid this problem, a threshold of 0.1 second or 800 samples is selected on the basis of the statistics of short words. By removing too close candidate word boundaries the false alarms were reduced, although it resulted in a small increase in the percentage of misses.

3.4 Hits and False Alarms

If a word boundary hypothesis matches the actual word boundary it is a hit (H) and if a word boundary hypothesis doesn't match are the actual word boundary it is called False Alarm (FA), the percentage of hits and false alarms is calculated using Equation 3.1 and Equation 3.2:

$$\text{Percentage of hits} = \text{hits/act} \quad \textbf{(3.1)}$$

$$\text{Percentage of FA} = \text{false alarms/candidates} \quad \textbf{(3.2)}$$

Where **hits** is the number of word boundaries hypothesized, **act** is total number of actual word boundaries, **false alarms** is number of erroneous hypothesis and **candidates** is total number of hypothesis. The actual word boundaries of speech files used from Switchboard database were observed to identify the threshold of trade off and a maximum trade off of 600 samples is considered as allowable distance between candidate word boundary and actual word boundary to consider it as hit. If the distance between a candidate word boundary and actual word boundary is more than 600 samples it is considered as false alarm. The threshold of 600 samples was chosen to match Switchboard database

3.5 Energy:

Energy is a basic acoustic feature of speech. Cues by observing the behavior of energy at word boundaries of speech were used. Initially speech files of 10 second length were used to identify the cues from energy to detect the word boundaries. After identifying the cues and developing an algorithm using

energy to detect word boundaries, it was tested on the utterances selected from Switchboard database which has three thousand word boundaries.

3.5.1 Cues from Energy:

The changes in energy at the word boundaries were observed to obtain the cues to identify them. From **Figure 3.1** it can be observed that, at the word boundaries, amplitude has sudden rises or falls. The increase in amplitude from the previous frame to the current frame is taken as the cue for word boundary, also a decrease in energy from the previous frame to the current frame is taken as cue to identify the word boundary. A threshold is identified for each utterance, which is mean of energy of the speech file, was used to determine the word boundaries.

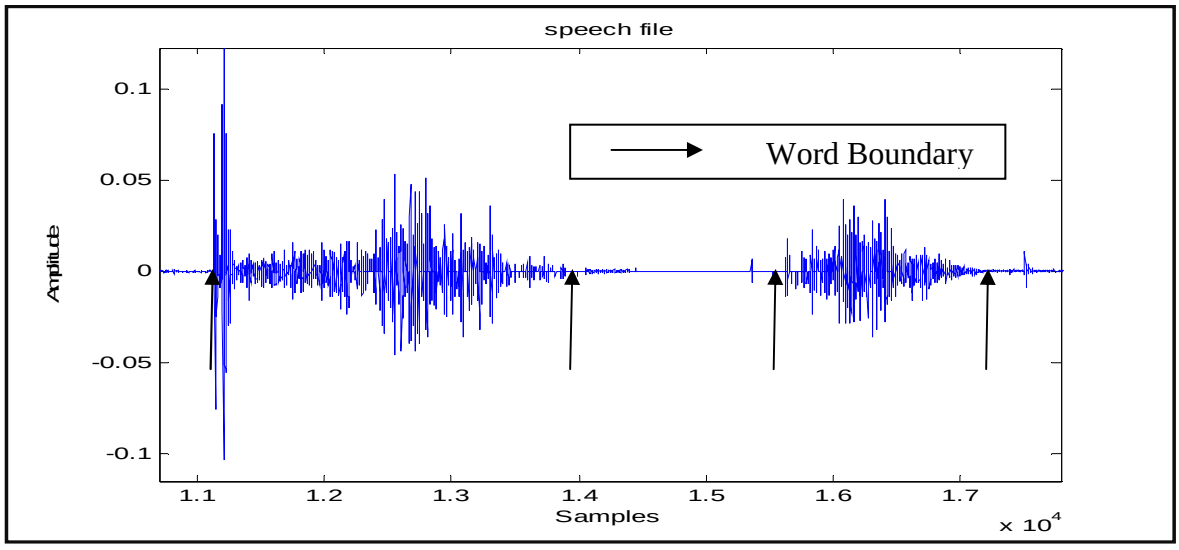

Figure 3.1: Utterance with two words with word boundaries identified using transcriptions

3.5.2 Energy Calculation

Each speech file is divided into frames of 20ms was used with 50% overlap to obtain the optimal results. Frame overlap proved to be very useful in detection of the word boundaries. After dividing the speech file into frames, energy of each frame is calculated using Equation 3.3:

$$E(n) = \sum_{i=1}^{k} s(i)^2 \quad \textbf{(3.3)}$$

Where **E(n)** is the energy of frame **n**, **s** is the speech signal and **k** is the frame length. The energy of each frame along with the speech file is shown in **Figure 3.2**

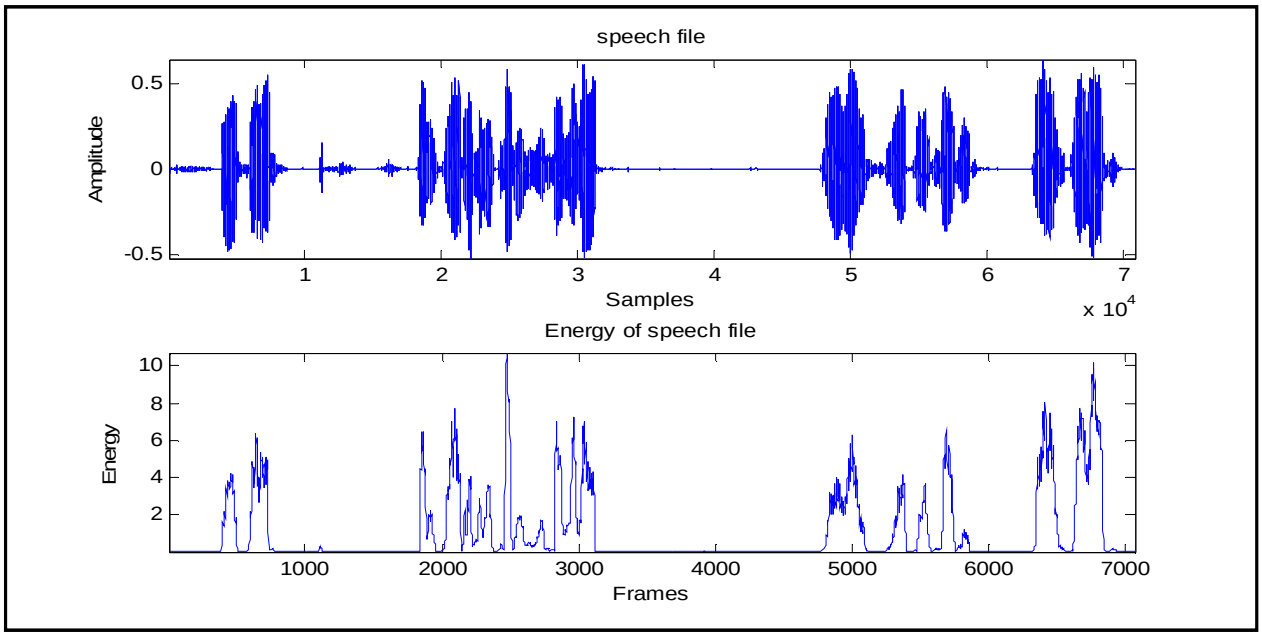

Figure 3.2: Speech file (top panel) and energy of speech file (bottom panel)

After identifying the energy of each frame, a threshold is identified for each speech file which is the mean of the energy of speech file. A sample in the speech file is identified as the word boundary based on the following criteria and equations:

Cue1: Word boundary has a rise in energy from the previous frame which can be the beginning of the word:

if (**E(i-2)<= t** and **E(i-1)<t** and **E(i)>=t** and **E(i+1)>t** and **E(i+2)>t**) , **(3.4)**

Where **i** is the possible place of the word boundary which is marked as a candidate word boundary, where **E(i)** is the energy of frame **i** and **t** is the threshold of energy

Cue2: Word boundary has fall in energy from the previous frame which can possibly be the end of the word:

if (**E(i-2)> t** and **E(i-1)>t** and **E(i)<=t** and **E(i+1)<t** and **E(i+2)<t**) , **(3.5)**

Where **i** is the possible place of the word boundary which is also called candidate word boundary, and **E(i)** is the energy of frame **i** and **t** is the threshold of energy

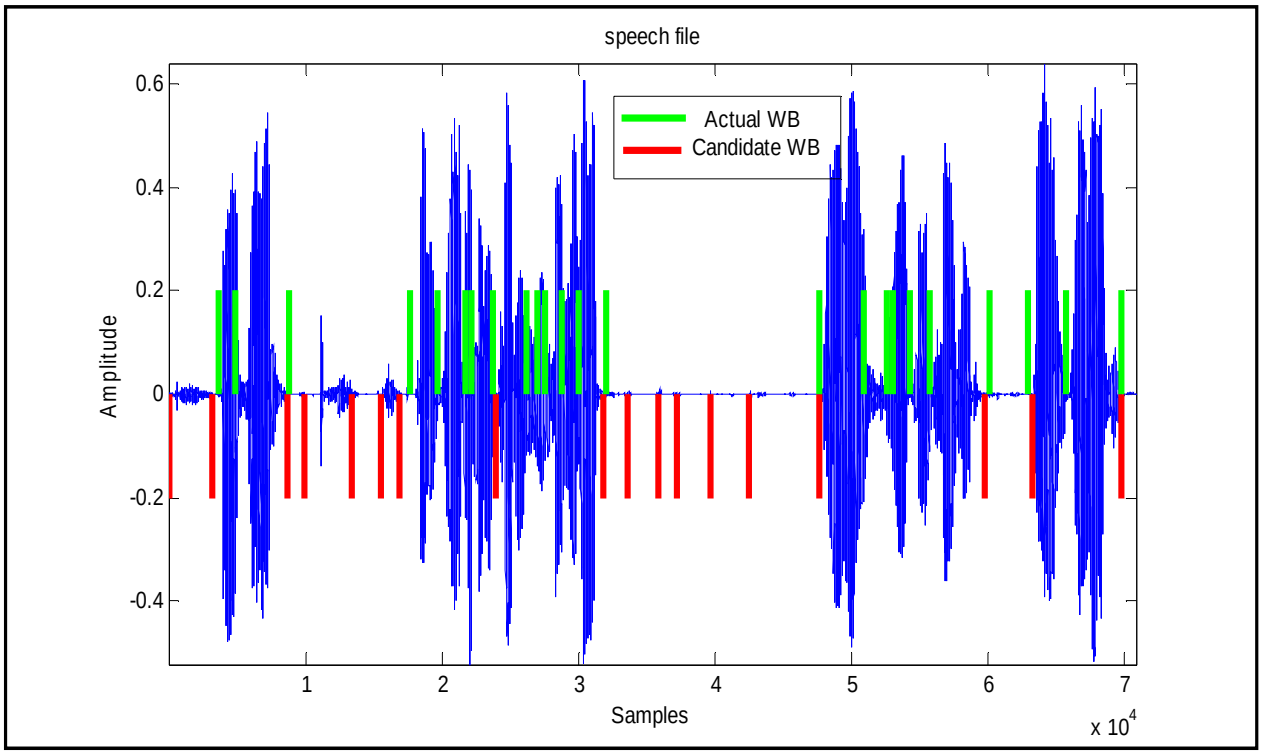

Figure 3.3: Figure shows the actual word boundaries (green) from Switchboard and candidate word boundaries (red) which were determined using energy

Also a threshold is applied on the distance between two candidate word boundaries to avoid the algorithm identifying candidate word boundaries that are too close as discussed in **Section 3.3**

3.5.3 Results Using Energy:

Detection rates and false alarms using energy are shown in **Table 3.1**. Three thousand word boundaries of Switchboard database were used in evaluating the performance of energy. Mean of energy is used as threshold for energy. Although 56 percent of hits were obtained using energy, the false alarms rate is very high. The performance of the energy is found to be good only if the words are separated by pause or silence. Many boundaries with no well defined pauses were missed, e.g. between points 2 and 3 in **Figure 3.3**, using energy.

Table 3.1: Results of Energy

Technique	Hits	False Alarms
Energy	56	40

3.5.4 Inference from Results

Strengths of energy in identifying the word boundaries are:

- It is very good at identifying the word boundaries with well defined word boundaries.
- It helps in confirming the predictions of the word boundaries using other techniques.
- The hits of energy are more closely identified to the actual word boundaries than with either Teager Energy or Pitch.

Drawbacks of energy in identifying the word boundaries are:

- The word boundaries are identified only if there is a well defined boundary. If there is no well defined boundary between two words, this technique results in misses.
- It also identifies some of the boundaries of the noise as word boundaries which results in false alarms.

3.6 Teager Energy:

Teager Energy estimates the true energy of the source of resonance. It has both frequency and amplitude energy information which is effective in identifying the word boundaries. In using Teager Energy to identify the word boundaries, the speech file need not be divided into frames like energy. It only operates on three consecutive samples and therefore it can be applied on the fly. It is also very effective in responding to the changes in both amplitude and frequency. Cues to identify the word boundaries are recognized initially using small speech files before proceeding to test it on longer speech files in the Switchboard database.

3.6.1 Cues from Teager Energy:

The changes in Teager Energy at the word boundaries were observed in order to obtain the cues used to identify word boundary. Teager Energy like energy has increasing magnitude at the beginning of the word and it has decreasing magnitude at the end of the word. From **Figure 3.4** it can be observed that at the word boundaries Teager Energy has sudden rises or falls in the amplitude at word boundaries. The changes in amplitude from the previous sample to the current frame is more effectively tracked by Teager Energy than energy and it is taken as the cue for beginning of the word boundary, also a decrease in Teager Energy from the previous frame to the current frame is taken as cue to identify the end of word boundary.

3.6.2 Teager Energy Calculation

Teager Energy operates only on three consecutive frames and is calculated using Equation 3.6:

$$TE(n) = s(n)^2 - s(n-1)*s(n+1) \qquad (3.6)$$

Where **n** is the index of the sample and **s** is the speech signal. Teager Energy is shown in **Figure 3.5**

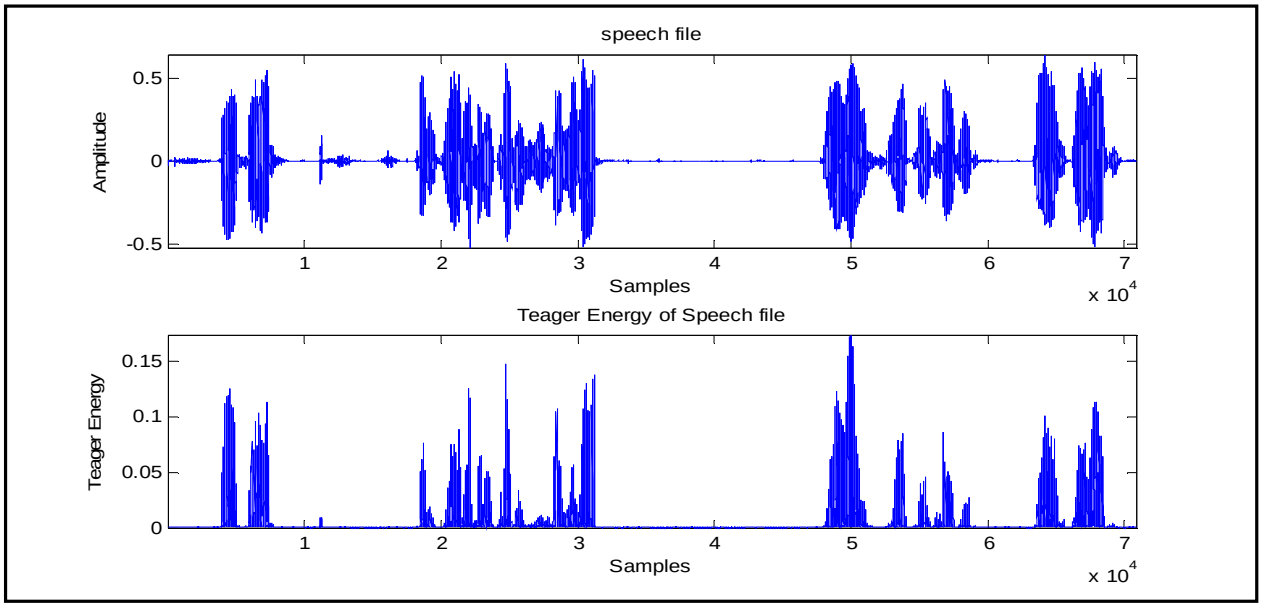

Figure 3.5: Speech file (top panel) and Teager Energy of speech file (bottom panel)

Teager Energy at each sample is calculated and a threshold is identified which is mean of the Teager Energy for each utterance. A sample in the speech file is identified as the word boundary based on the following cues and equations.

Cue1: Word boundary has rise in Teager Energy from the previous sample which can be the beginning of the word:

if (**TE(i-2)<= t** and **TE(i-1)<t** and **TE(i)>=t** and **TE(i+1)>t** and **TE(i+2)>t**) , (3.7)

Where **i** is the possible place of the word boundary which is also called candidate word boundary, **TE(i)** is the Teager Energy of a sample **i** and **t** is the threshold of Teager Energy

Cue2: Word boundary has fall in Teager Energy from the previous sample which can be the end of the word:

if (**TE(i-2)>t** and **TE(i-1)>t** and **TE(i)>=t** and **TE(i+1)<t** and **TE(i+2)<t**) , (3.8)

Where **i** is the possible place of the word boundary which is also called candidate word boundary, where **TE(i)** is the Teager Energy of a sample **i** and **t** is the threshold of Teager Energy.

3.6.3 Results Using Teager Energy:

Detection rates and false alarms using Teager Energy are shown and compared with energy in **Table 3.2**. Three thousand word boundaries of Switchboard database were used in evaluating the performance of energy. Mean of Teager Energy is used as threshold.

Table 3.2: Results of Teager Energy and Energy

Technique	Hits	False Alarms
Teager Energy	72	27
Energy	56	40

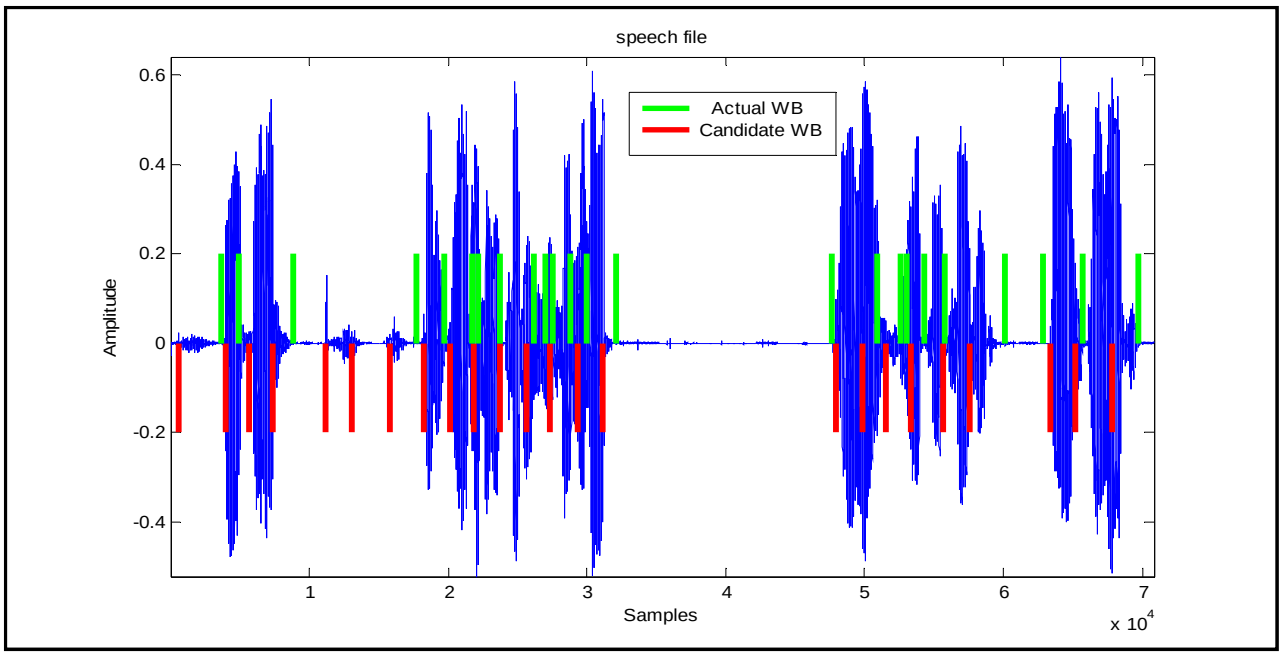

Figure 3.6: Figure shows the actual word boundaries (green) from Switchboard and candidate word boundaries (red) which were plotted using the cues from Teager Energy

Teager Energy produced better results than energy in terms of hits and false alarm rates were much less when compared with energy.

3.6.4 Inference from Results

Strengths of Teager Energy in identifying the word boundaries are:

- It is able to identify the word boundaries even if there are no well defined boundaries between the words, e.g., between points 2 and 3 in **Figure 3.6**
- Speech is non-stationary and the acoustic features change very rapidly, Teager Energy is capable of responding to changes in both amplitude and frequency very rapidly.

Drawbacks of Teager Energy in identifying the word boundaries are:

- Teager Energy like energy gives false alarms identifying the boundaries of noise as word boundaries

- If the ending of the word is extended due to the prosody of the speaker, Teager Energy is not able to identify these word boundaries close to the actual word boundaries.

3.7 Pitch:

Pitch is the inverse of fundamental frequency of a signal and it is one of the acoustic features of speech. The fundamental frequency exists only for voiced sounds. Pitch is used as a cue for identifying the word boundaries. English is a stress bound language. This feature of English is used as another cue for identifying word boundaries using pitch. A pitch detection algorithm Sawtooth Waveform Inspired Pitch Estimator (SWIPE) by Camacho *et al* [2007] was used to estimate pitch of each file and the pitch from SWIPE was used in detecting the word boundaries. The goal of SWIPE is to find the frequency that maximizes the peak to valley ratio in each frame of the speech file. Several modifications to this idea were applied to improve its performance: the locations of the harmonics of each frame were blurred, the spectral amplitude and the frequency scale were warped, an appropriate window type and size were chosen, and simplifications to reduce computational cost were introduced, which makes the algorithm robust. Swipe is compared with fourteen different techniques, including autocorrelation, cross correlation, YIN, and is ranked as second best technique for pitch estimation.

3.7.1 Pitch Estimation Using SWIPE:

Sawtooth Waveform Inspired Pitch Estimator (SWIPE) estimates pitch using the following steps:

- Compute the square-root of the spectrum of the signal for every 10ms.
- Normalize the square-root of the spectrum and apply an integral transform using a normalized cosine kernel whose envelope decays as $1/\sqrt{f}$, where f is a pitch candidate identified by SWIPE.
- Estimate the pitch as the candidate with highest strength.

Swipe also limits the range of pitch between 75 and 500 Hz which is the range of pitch for speech. It estimates pitch every 0.01 second. **Figure 3.7** shows the estimate of pitch using Swipe and the pitch breaks, between 100 and 200 in the bottom panel of **Figure 3.7,** were used as cues in identifying the candidate word boundaries

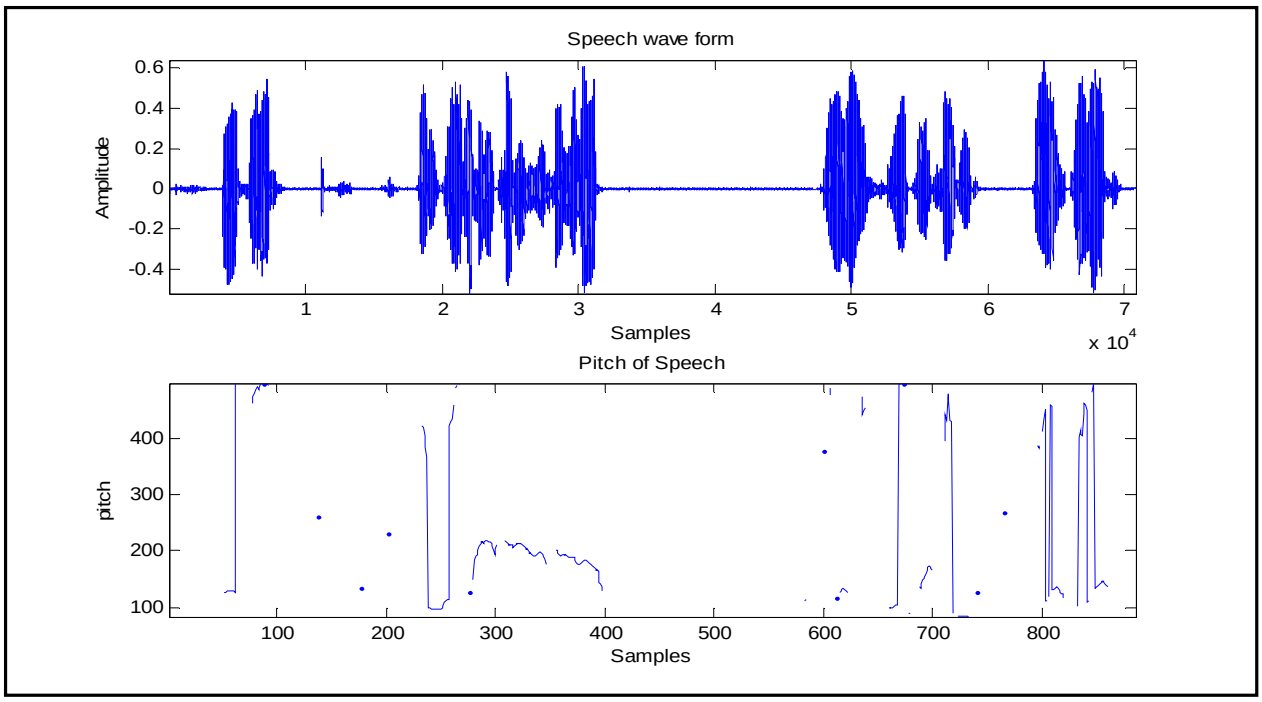

Figure 3.7: Speech file (top panel) and pitch estimate of speech file using SWIPE (bottom panel)

3.7.2 Detecting Word Boundaries Using Pitch:

The following cues are used to estimate pitch using SWIPE to detect word boundaries:

Cue1: Pitch of the previous frames does not exist or is much less than the current frame if the current frame is the beginning of a word boundary.

if (**p(i-2) = 0** and **p(i-1)=0** and **p(i) ≠ 0** and **p(i+1) ≠ 0** and **p(i+2) ≠ 0**) , (3.9)

Where **i** is the possible place of the word boundary which is also called candidate word boundary and **p(i)** is the pitch of frame **i.**

Cue2: Pitch of the next frames does not exist or is much less than the current frame if the current frame is the beginning of a word boundary.

if (**p(i-2) ≠ 0** and **p(i-1) ≠ 0** and **p(i) =0** and **p(i+1) =0** and **p(i+2)=0**), (3.10)

Where **i** is the possible place of the word boundary which is called candidate word boundary and **p(i)** is the pitch of frame **i**. Also a threshold is applied on the distance between two candidate word boundaries to avoid the algorithm to identifying close candidate word boundaries which are close as discussed in **Section 3.3.**

3.7.3 Results Using Pitch:

Detection rates and false alarms using pitch are shown and compared with Teager Energy and energy in **Table 3.3**. Three thousand word boundaries of Switchboard database were used in evaluating the performance of pitch. Using pitch 68 percent of the word boundaries were identified which is better than energy and Teager Energy performed better than pitch. The false alarms rate is also high using pitch. However Pitch detected the word boundaries more accurately than Teager Energy when the word boundaries are prolonged by the speaker due to prosody, e.g. at point 6 in **Figure 3.8.** Some word boundaries with no well defined pauses, e.g., between points 2 and 3 in **Figure 3.8**, were detected using pitch which were missed by energy.

Table 3.3: Results of Pitch, Teager Energy and Energy

Technique	Hits	False Alarms
Pitch	68	29
Teager Energy	72	27
Energy	56	40

3.7.4 Inference From Results

Strengths of pitch in identifying the word boundaries are:

- It is capable of reducing the false alarms by not identifying the boundaries of noise as word boundaries like Teager Energy and energy.

- It is able to identify the beginning of the word boundaries effectively, as English is a stress bound language.

Drawbacks of pitch in identifying the word boundaries are:

- There is some offset between actual and predicted word boundaries if the word is beginning with unvoiced speech.
- Some false alarms in this technique occur due to the pitch reset due to the prosodic units in the same word.

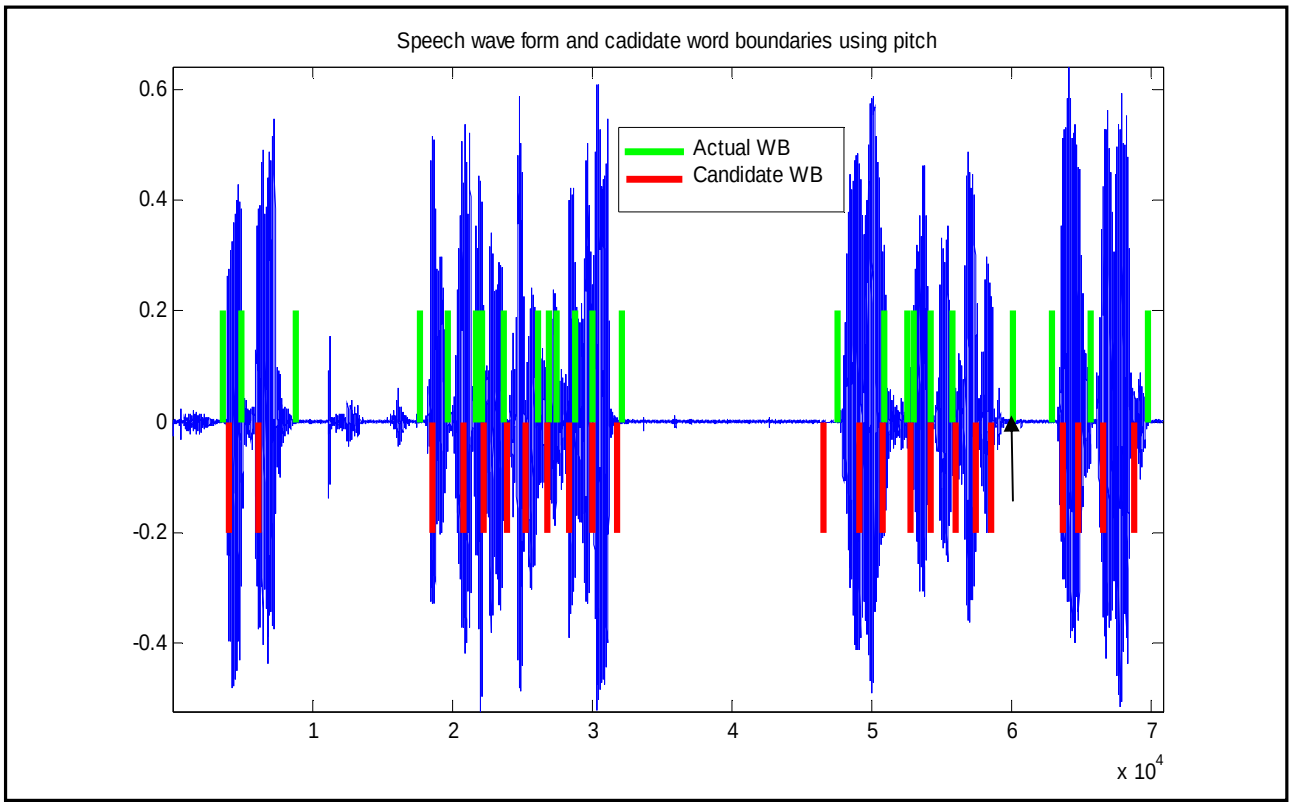

Figure 3.8: Figure shows the actual word boundaries (green) from Switchboard and candidate word boundaries (red) which were plotted using the cues from pitch

3.8 Fusion of All the Techniques:

After identifying the candidate word boundaries of all three techniques separately, the decisions (candidate word boundaries) from all the techniques were processed in a data fusion module to make a final decision. Simple counting rule for optimal data fusion developed by Mansouri and Fathi (2003) was used to fuse the decisions from the three techniques. The major advantage of the counting rule is that *a prior* knowledge of the probability mass functions is not required. Knowledge about the reliability of each of the techniques is acquired by the model based on reinforcement learning.

In this research the decisions from three techniques are input into the fusion model. Different weights, including equal weights, were used to obtain the optimal hit rate. When the initial weights are 1 for energy and pitch, and 2 for Teager Energy gave best results shown in **Table 3.5**. Weights are then iteratively updated according to Equation 3.11 and Equation 3.12:

$$w_{i+1} = w_i + 1/m_i \qquad \text{if } u_i = d_i \qquad (3.11)$$

$$w_{i+1} = w_i - e^{w_i}/m_i \qquad \text{if } u_i \neq d_i \qquad (3.12)$$

Where $\mathbf{w_{i+1}}$ is the weight of the detector or technique for next iteration, $\mathbf{m_i}$ is the number of decisions made by $\mathbf{i^{th}}$ detector that agree with the final decision $\mathbf{d_i}$ made by the fusion model and $\mathbf{u_i}$ is the decision of the i^{th} local detector. The iterations are repeated until a steady state value of the weights is reached. The decision, if a candidate word boundary is a boundary or not a boundary, is made based on Equations 3.13 and 3.14:

$$u = 1 \qquad \text{if } \sum_{j=1}^{n} w(i)u(i) > t \qquad (3.13)$$

$$u = -1 \qquad \text{otherwise} \qquad (3.14)$$

Where **u = 1** means that the candidate word boundary is a boundary and **u = -1** means that candidate word boundary is not a word boundary, **t** is threshold for reliability and in this research, after using different numbers and comparing the results, **t** is considered as 3 for optimal results. Initial weights assigned to each of the technique are iteratively updated, based on the decisions made by each of the

technique and the decision made by the counting rule, using Equations 3.11 and 3.12. **Table 3.4** shows the weights of the techniques updated by the counting rule for one of the speech files. Note - Energy has less weight as its hit rate is less and false alarm rate is more. Teager Energy has the highest weight in almost all the cases, shown in **Table 3.4**, as it has the highest hit rate and less of a false alarm rate than the other two techniques. The system is said to reach steady state if the difference in weights of two consecutive iterations is less than certain threshold, which is set to 0.05 for optimal results.

Weights were estimated using counting rule and a steady state is reached approximately after 2500 iterations as shown in **Figure 3.9**. The weights estimated were used to fuse the decisions of the three techniques, energy, Teager Energy and pitch. Using the updated weights from counting rule has improved the performance of the system to 79% which was 73% by using the linear opinion pool.

Table 3.4: Weights assigned by counting rule

Technique	Energy	Teager Energy	Pitch
Initial weights	1	2	1
Weight Assigned by Counting Rule	1.476	2.407	1.825

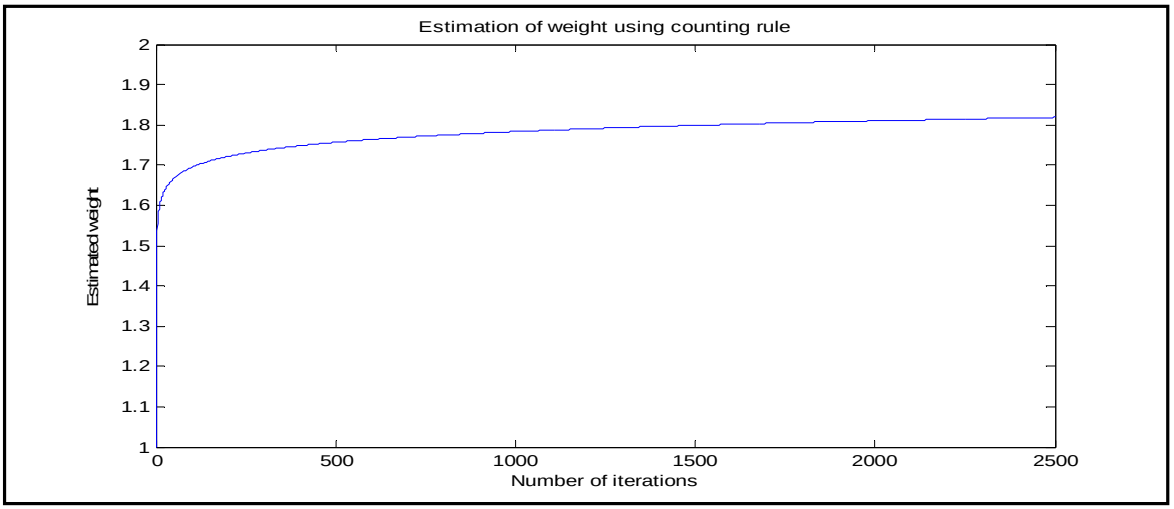

Figure 3.9: Weight of pitch updated using counting rule for every iteration

3.8.1 Results:

The threshold for avoiding two close candidate word boundaries was reduced to 600 samples (which was 800 previously) while using the counting rule to fuse the decisions from each of the techniques. This is done to increase the percentage of hits which also increases the false alarms. The counting rule is based on updated weights, and was able to remove many false alarms while retaining approximately 90 percent of the hits from each of the techniques. **Figures 3.10, 3.11, 3.12** shows the candidate word boundaries (green) from each of the techniques which have increased number of candidate word boundaries compared to **Figures 3.3, 3.6 and 3.8**, and also increased false alarms. **Figure 3.13** shows the decisions made by counting rule after fusing data from all of the three techniques.

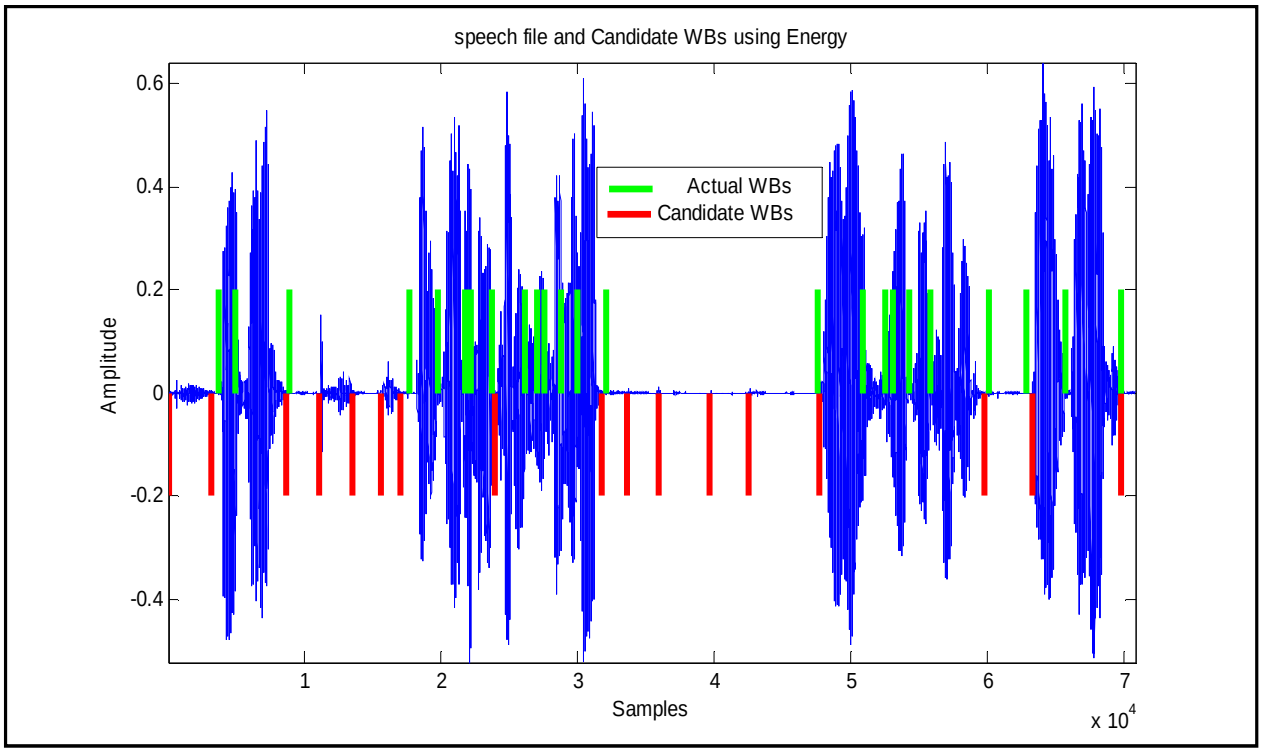

Figure 3.10: Figure shows the actual word boundaries (green) of Switchboard data and candidate word boundaries (red) which were plot determined using Energy

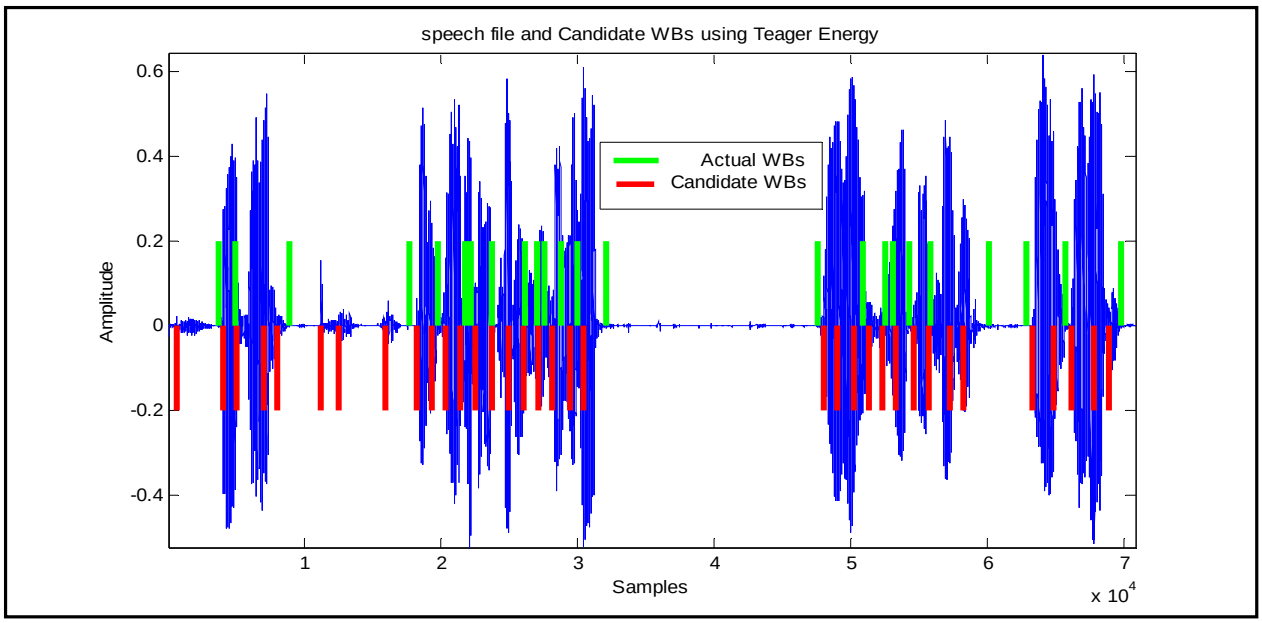

Figure 3.11: Figure shows the actual word boundaries (green) of Switchboard data and candidate word boundaries (red) which were determined using Teager Energy

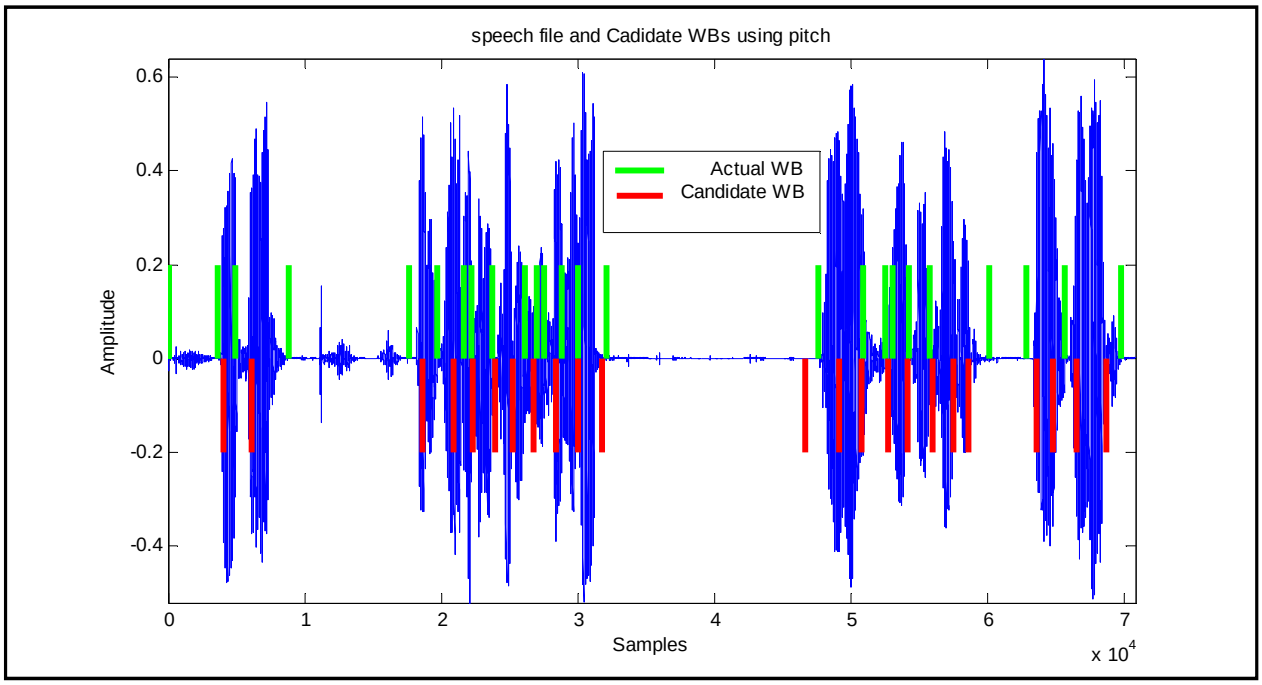

Figure 3.12: Figure shows the actual word boundaries (green) of Switchboard data and candidate word boundaries (red) which were determined using Pitch

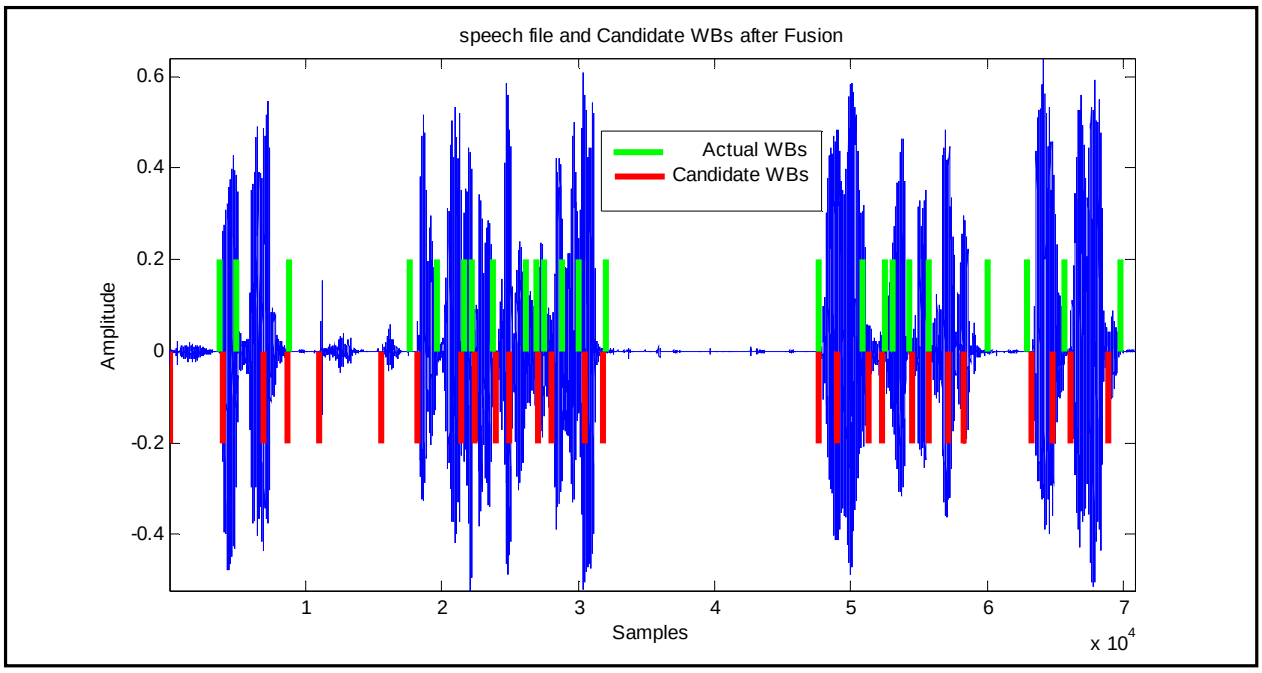

Figure 3.13: Figure shows the actual word boundaries (green) of Switchboard data and candidate word boundaries (red) which were determined by counting rule

Fusion using counting rule with equal initial weight of 1 for all the techniques was evaluated and counting rule with initial weight of 1 for energy, 2 for Teager energy and 1 for pitch was evaluated and results are shown in **Table 3.5**.

Table 3.5: Results from counting rule

Techniques	Hits %	False alarms %
Energy	49	57
Teager Energy	80	42
Pitch	81	55
Counting Rule with unequal initial weights	79	27
Counting Rule with equal initial weights	77	32

Database: Switchboard; Speech type: Spontaneous; Number of Word Boundaries: 3000 (approximately)

Table 3.6: Results from previous publications

S. No	Publication name	Words/ sentences used	Percentage of hits	Percentage of false alarm
1	Ramana Rao [1996]	50-sentences, 5 to 15 words per sentence and 10 speakers, not spontaneous	85 for four Indian languages 65 for German	28
2	Rajendran [1996]	50 sentences read by 5 native speakers 587 word boundaries	74	14
3	Iwano and Hirose [1999]	Used Mora language and total words is 326 Not spontaneous	77	15
4	Agarwal and Jain [2010]	3- speakers 750- words Non- spontaneous	72 to 93	14 to 31

Linear Opinion Pool (LOP) was also used to fuse, by assigning equal weights predefined as 1 to all the techniques, the decisions from the three techniques and the results are shown in **Table 3.7**. Log opinion pool was used with different weights assigned, 1 for energy and pitch and 2 for Teager Energy, based on performance of each technique individually. Counting rule performed much better than the linear opinion pool and log opinion pool as shown in **Table 3.7**.

Table 3.7: Comparison of counting rule, linear opinion pool and log opinion pool

Technique	Hits %	False Alarms %
Linear Opinion Pool	73	34
Log Opinion Pool	74	29
Counting Rule	79	27

Database: Switchboard; Speech type: Spontaneous; Number of Word Boundaries: 3000 (approximately)

3.8.2 Inference From Results:

The hits percentage was allowed to increase at the cost of false alarms. The weights of each of the techniques shown in **Table 3.4** were modified iteratively by the counting rule. Although the false alarms were high in each of the techniques as shown in **Table 3.5**, the counting rule was able to remove the false alarms with minimum loss of hits. It is shown in **Table 3.6** that although different techniques to identify the word boundaries performed well, the database used by them was not spontaneous, which is a major challenge in speech processing. In this research only spontaneous speech was used from Switchboard database. Even with spontaneous speech, the three techniques Energy, Teager Energy, Pitch and the fusion model, the counting rule yielded good percentage of hits.

CHAPTER 4

FUTURE WORK

4.1 Keyword Spotting

Secondary goal of this research is keyword spotting. Keyword spotting is the task of identifying the presence of the desired words in an arbitrary speech. Three different approaches shown in the **Figure 4.1** can be used to spot keywords.

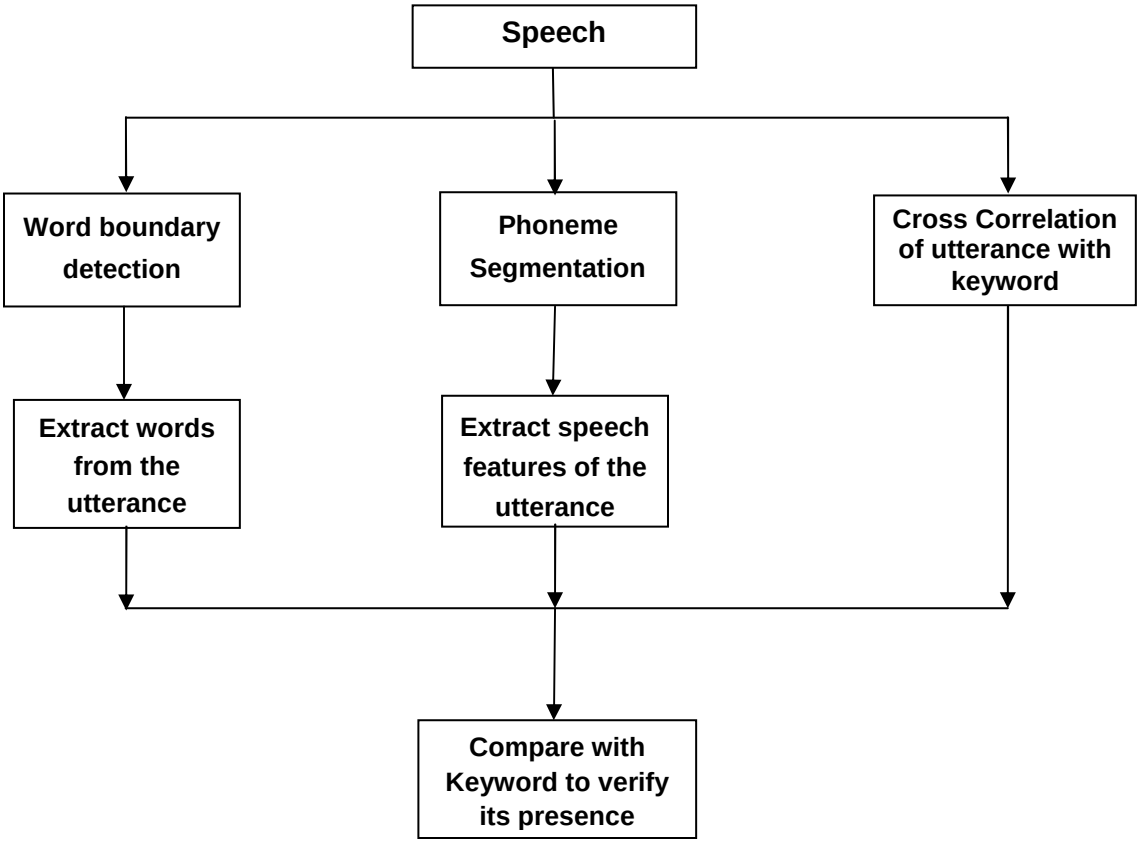

Figure 4.1: Different approaches to keyword spotting

After identifying the word boundaries, the words are extracted from the utterance. Short words, which are of a shorter length than the keyword, can be removed to reduce the search space. The features of the words extracted are them compared with the keywords to verify the presence. Keyword spotting can also be achieved by phoneme segmentation. The utterance is segmented into phonemes and the features extracted are compared with the features of the keyword to verify its presence. Cross correlation measures the similarity between two signals. This can be used to measure the similarity between the keyword and the utterance to spot the keywords.

4.2 Word Boundary Detection

A word boundary detection algorithm for Hindi language using the prosodic features like pitch and intensity was implemented by Agarwal and Jain *et al* [2010]. Cues about the behavior of intensity and pitch are observed to detect the word boundaries in Hindi language. Behavior of intensity and pitch are used in conjunction to detect the candidate word boundaries rather than fusing the candidates from each of the techniques. The algorithm has a highest hit rate of 98% and least of 72% with a false alarm rate of 14% to 31%.This approach also helps in reducing the false alarms. Intensity can also be used along with the three techniques used in this research to identify the word boundaries and the data can be fused for more reliability. The knowledge of acoustic features in particular voiced and unvoiced segment plays an important role in many speech analysis systems.

Accurate endpoint detection is crucial for good speech recognition accuracy. Energy-based endpoint detection algorithms are computationally efficient and perform adequately in quiet conditions but they have little to no immunity to background noise. Sahar and Khaled *et al* [2002] proposed a new approach relying on the energy and spectral characteristics to determine whether a frame is speech or non-speech. The main idea is to characterize the background or silence at the beginning of an utterance with discriminating statistics and then relying on these statistics to classify speech vs. non-speech. The speech vs. non-speech classification relies on two measures:

1. The current frame energy relative to the average background/silence energy level, and
2. Euclidean cepstral distance between the cepstral vector of the current frame and the mean cepstral vector representing the background/silence.

For a speaker-independent recognizer at 20 dB SNR, it improves the false rejection rates by 45% for babble noise, and 29% for both office and lobby noise. Such a classification can be useful in word boundary detection which can increase the accuracy of the word boundary detection algorithm by rejecting the non-speech frames.

Segmentation of speech into voiced, unvoiced and phonemes can be used to identify the word boundaries. Finding the cues of word boundaries using segmentation and fusing the decisions along with energy, Teager Energy and pitch can improve the detection rates of the algorithm.

REFERENCES

1. Kaiser, J.F.; "On a simple algorithm to calculate the 'energy' of a signal" Conference on Acoustics, Speech, and Signal Processing, Page(s): 381 – 384, 1990
2. Ying, G.S.; Mitchell, C.D.; and Jamieson, L.H.; "Endpoint detection of isolated utterances based on a modified Teager Energy measurement" IEEE International Conference on Acoustics, Speech, and Signal Processing, Page(s): 732 – 735, 1993.
3. Junqua, J.-C.; Mak, B.; and Reaves, B.; "A robust algorithm for word boundary detection in the presence of noise," IEEE Transactions on Speech and Audio Processing, vol. 2, Page(s): 406–412, February 1994.
4. Ramanarao, G.V.; Srichland, J.; "Word boundary detection using pitch variations" Conference on Spoken Language, 1996. ICSLP 96. Proceedings., Page(s): 813 – 816, 1996
5. Rajendran, S.; Yegnanarayana, B.; "Word boundary hypothesization for continuous speech in Hindi based on F0 patterns" Speech Communication, Page(s): 21-46, Jan. 1996
6. Agaiby, H.; Moir, T.J.; "A robust word boundary detection algorithm with application to speech recognition" 13th International Conference on Digital Signal Processing Proceedings, Page(s): 1179 – 1182, 1997.
7. McKinley, B.L.; Whipple, G.H.; "Model based speech pause detection" IEEE International Conference on Acoustics, Speech, and Signal Processing, Page(s): 1179 – 1182, 1997.
8. Cettolo, M.; Falavigna, D.; "Automatic detection of semantic boundaries based on acoustic and lexical knowledge," 5th International Conference on Spoken Language Processing, 1998.
9. Kittler, J., Hatef, M., Duin, R. P. W., and Matas, J., "On Combining Classifiers', IEEE Trans. Pattern Anal. and Mach. Intell., Vol. 20, No. 3, Page(s):226-239, Mar. 1998.
10. Punska, O.; "Bayesian Approaches to Multi-Sensor Data Fusion" A dissertation submitted to the University of Cambridge, 1999.
11. Iwano, K.; Hirose, K.; "Prosodic word boundary detection using statistical modeling of moraic fundamental frequency contours and its use for continuous speech recognition" IEEE

International Conference on Acoustics, Speech, and Signal Processing, Page(s): 541 – 554, 1999.

12. Wu, G.; Lin, C.;"Word boundary detection with mel-scale frequency bank in noisy environment" IEEE Transactions on Speech and Audio Processing, Page(s): 541 – 554, 2000.
13. Hirose, K.; Iwano, K.; "Detection of prosodic word boundaries by statistical modeling of mora transitions of fundamental frequency contours and its use for continuous speech recognition" IEEE International Conference on Acoustics, Speech, and Signal Processing, Page(s): 1763 – 1766, 2000.
14. Jefremov, A.; Kleijn, W.B.; "Spline-based continuous-time pitch estimation" IEEE International Conference on Acoustics, Speech, and Signal Processing, Page(s): 337–340, 2002.
15. Wang, D.; Lu, L.; and Zhang, H.; "Speech segmentation without speech recognition" Acoustics, Speech, and Signal Processing, 2003. Proceedings. 2003 Page(s): 89–101, March 2002.
16. Bou-Ghazale, S.E.; Assaleh, K.; "A robust endpoint detection of speech for noisy environments with application to automatic speech recognition" IEEE International Conference on Acoustics, Speech, and Signal Processing, Page(s): 3808-3811, 2002.
17. Mansouri, N.; and Fathi, M.; "Simple counting rule for optimal data fusion", Proceedings of 2003 IEEE Conference on Control Applications. Page(s): 1186 – 1191, 2003.
18. Yantorno, R.E.; Smolenski, B.Y.; and Chandra, N.; "Usable speech measures and their fusion" Proceedings of the 2003 International Symposium on Circuits and Systems, 2003.
19. Kahler, O.; Denzler, J.; and Triesch, J.; "Hierarchical sensor data fusion by probabilistic cue integration for robust 3D object tracking" 6th IEEE Southwest Symposium on Image Analysis and Interpretation, Page(s): 216 – 220, 2004.
20. Wang, D.; Narayanan, S.S.; "A multi-pass linear fold algorithm for sentence boundary detection using prosodic cues" IEEE International Conference on Acoustics, Speech, and Signal Processing, Page(s): 525-528, 2004.
21. Hui, L.; Dai, L.; and Wei, L.; "A pitch detection algorithm based on AMDF and ACF" Conference on Acoustics, Speech and Signal Processing, 2006.

22. Arifianto, D.; “Dual Parameters for Voiced-Unvoiced Speech Signal Determination” Acoustics, IEEE International Conference on Speech and Signal Processing, Page(s): 749 – 752, 2007.
23. Resch, B.; Nilsson, M.; Ekman, A.; and Kleijn, B.;” Estimation of the Instantaneous Pitch of Speech” IEEE Transactions on Audio, Speech, and Language Processing, Page(s): 813 - 822, 2007.
24. Camacho, A.; “SWIPE: A Sawtooth Waveform Inspired Pitch Estimator For Speech And Music” dissertation presented to the Graduate School of the University of Florida, 2007.
25. Xiaokun, L.; Deng, Y.; “Combining Speech Energy and Edge Information for Fast and Efficient Voice Activity Detection in Noisy Environments” Conference on Pattern Recognition, Page(s): 1 – 4, 2008.
26. Xufang, Z.; O'Shaughnessy, D.; “A new hybrid approach for automatic speech signal segmentation using silence signal detection, energy convex hull, and spectral variation” Conference on Electrical and Computer Engineering, Page(s): 145 – 148, 2008.
27. Tsiartas, A.; Ghosh, P.K.; Georgiou, P.; and Narayanan, S.; “Robust word boundary detection in spontaneous speech using acoustic and lexical cues” IEEE International Conference on Acoustics, Speech and Signal Processing, Page(s): 4785 – 4788, 2009.
28. Audhkhasi, K.; Kandhway, K.; Deshmukh, O.D.; and Verma, A.; “Formant-based technique for automatic filled-pause detection in spontaneous spoken english ” IEEE International Conference on Acoustics, Speech and Signal Processing, Page(s): 4857 – 4860, 2009.
29. Nehe, N.S.; Holambe, R.S.; “Mel Frequency Teager Energy Features for Isolate Word Recognition in Noisy Environment” Conference on Emerging Trends in Engineering and Technology (ICETET), Page(s): 904 – 908, 2009
30. Verteletskaya, E.; Sakhnov, K.; and Simak, B.; “Pitch detection algorithms and voiced/unvoiced classification for noisy speech” 16th International Conference on Systems, Signals and Image Processing, Page(s): 1 – 5, 2009.
31. Nuno, Pratas.; Nicola M.; Prasad, N.R.; Rodrigues, A.; Prasad R.; “Robust Cooperative Spectrum Sensing for Disaster Relief Networks in Correlated Environments” Journal of Selected Areas in Communications - Advances in Cognitive Radio Networking and Communications, 2009.

32. Chen, Y.; Tian, S.; and Sun, B.; “Decision Fusion for Structural Damage Detection: Numerical and Experimental Studies” Hindawi Publishing Corporation, Advances in Civil Engineering, 2010.
33. Agarwal, A.; Jain, A.; Prakash, N.; and Agrawal, S.S.; “Word boundary detection in continuous speech based on suprasegmental features for Hindi language” 2nd International Conference on Signal Processing Systems (ICSPS), Page(s): 591-594, 2010.

Appendix

Matlab code

Main file for fusing decisions from energy, Teager Energy and pitch:

```
clear all
close all
clc
[x,fs] = wavread('C:\Users\Sandeep\Documents\Research\Codes\Sandeep Codes\new
codes\c.wav');
l = length(x);
de = zeros(l,1); %detection vector for energy
dt = zeros(l,1); %detection vector for Teager Energy
dp = zeros(l,1); %detection vector for pitch
m = mean(x);
m = abs(m);
%moving average filter
a=1;
h = 0.5;
b=[h h h h];
x = filter(b,a,x);

% reading the transcriptions
fid = fopen('C:\Users\Sandeep\Documents\Research\Codes\Sandeep Codes\new
codes\sw2273A-ms98-a-word.text','r');
data = textscan(fid,'%s %f %f %s');
%close file
fclose(fid);
start_time = data{2};
start_time = start_time(1:29); % change the length based on the length of the
speech file
start_sample = start_time*fs;
lf= length(start_time);
word_data = data{4};
word_data = word_data(1:lf);

j=1;
k(j)=0;
%identifying the silence time and removing it
for i=1:lf
    f1 = strcmp( word_data(i), '[silence]'); %local variable
    f2 = strcmp( word_data(i), '[noise]');

    if (f1==1 || f2==1)
         k (j+1)=i;
         j = j+1;
         if (k(j)-k(j-1)==1)
            start_sample(i)=0;
         end
     end
end
```

```
start_sample = start_sample';

ip = pitch_wb(x,fs,lf,start_sample);
it = teager_wb(x,fs,m,lf,start_sample);
ie = energy_wb(x,fs,m,lf,start_sample);
%ie = ones(1,15);
ip = ip';
it = it';
%ie = ie';

lt = length(it);
le = length(ie);
lp = length(ip);
ix = ie;
iy = it;
iz = ip;

%adusting the indexes from different techniques
ze = 0;
zt = 0;
for i = 1:le

    vt(1:lt,1)=ix(i);
    vp(1:lp,1)=ix(i);

    difference1 = vt-it;
    difference1 = abs(difference1);
    [ct,j] = min(difference1);

    if (ct<700)
    iy(j) = ix(i);
     ze = ze+1;
    end

    difference2 = vp-ip;
    difference2 = abs(difference2);
    [cp,k] = min(difference2);

    if (cp<600)
    iz(k) = ix(i);
     zt = zt+1;
    end


end

zp = 0;
for i = 1:lp

    vz(1:lt,1)=iz(i);
    difference1 = vz-it;
    difference1 = abs(difference1);
    [ct,j] = min(difference1);
```

```
    if (ct<500)
    iz(i)= iy(j) ;
    zp = zp+1;
    end

end
de(ix) = 1;
dt(iy) = 1;
dp(iz) = 1;

wp = 1;
we = 1;
wt = 2;
mt = 0;
me = 0;
mp = 0;
mtk(1) =0;
mek(1) =0;
for i = 1:2500 %optimization of fusion

    w = wp*dp+ we*de + wt*dt ;
    in = find(w>2);
    lef = length(in);
    for j = 1:lt
        vt1(1:lef,1)=iy(j);
        difference1 = vt1-in;
    difference1 = abs(difference1);
    [ct,k] = min(difference1);
    hk(j) = ct;
    if(ct==0)
        mt=mt+1;

    end
    end

    for j = 1:le
        ve1(1:lef,1)=ix(j);
        difference1 = ve1-in;
    difference1 = abs(difference1);
    [ce,k] = min(difference1);
    if(ce==0)
        me=me+1;

    end
     end

    for j = 1:lp
        vp1(1:lef,1)=iz(j);
        difference1 = vp1-in;
    difference1 = abs(difference1);
    [cp,k] = min(difference1);
    if(cp==0)
        mp=mp+1;
```

```
    end
    end

    mtk(i+1) = mt;
    mek(i+1) = me;
    mpk(i) = mp;

    if(mtk(i)<mtk(i+1))
    wt = wt + (1/mt);
    else
        wt = wt - (exp(wt)/mt);
    end

    if(mek(i)<mek(i+1))
    we = we + (1/me);
    else
        we = we - (exp(we)/me);
    end

    if(mpk(i)<mpk(i+1))
    wp = wp + (1/mp);
    else
        wp = wp - (exp(wp)/mp);
    end

     wtk(i) = wt;
     wek(i) = we;
     wpk(i) = wp;
end

li = length(in);

figure (4)
plot(x)
xlabel('Samples')
ylabel('Amplitude')
title('speech file and Candidate Wb s after Fusion')
hold on
axis tight
for i = 1:li

    figure(4)
   line([in(i) in(i)],[0 -0.2],'color','r','LineWidth',3)

end

for j = 1:lf
    figure(4)
    line([start_sample(j) start_sample(j)],[0 0.2],'color','g','LineWidth',3)
end


h = 0;
trade_off = 800;
```

```
int = in';
for i= 1:lf
    vectorz(1:li)= start_sample(i);
    difference = int-vectorz;
    difference = abs(difference);
    [c,j] = min(difference);
    c = abs(c);
    if (c < trade_off && start_sample(i)~=0)
    h = h+1;
    end
end

hits_percentage_fusion = h/li*100 ;%#ok<*NOPTS>
act = length(find(start_sample)) %number of actual word boundaries.

len_words = diff(start_time);
short_words = length(find(len_words<0.1 & len_words>0));
all_candidates = lt+lp+le-zp-ze-zt; %total candidates from all the techniques


fprintf('fusion results all candidates from 3 techniques = %4d \n hits = %4d
total cadidates = %4d  short words = %3d \n ',all_candidates
,h,li,short_words)
xlswrite('thesis_results', [act, all_candidates ,h,li,short_words,lf],'fusion
results','B19:G19')
```

Word boundary detection using pitch:

```
function [ip,fs] = pitch_wb (x,fs,lf,start_sample)

%calling the swipep function and estimating pitch
[p,~,s] = swipep(x,fs,[75 500],0.01,[],1/20,0.5,0.2);
pd = diff(p);
len= length(pd);
pd(len+1)=0;
k = find(real(pd)~=pd);
l1= length(k);


for i= 1:l1
    pd(k(i)) = 0;
end
e= pd;

% figure(3)
% plot(x)
% xlabel('Samples')
% ylabel('Amplitude')
% axis tight;
% title('Speech wave form and cadidate word boundaries using pitch')
% %
```

```
%
%
% for j = 1:lf
%     figure(3)
%     line([start_sample(j) start_sample(j)],[0
0.2],'color','g','LineWidth',3)
% end


threshold = 12; %threshold to avoid very short distant candidate word
boundaries
cut_off = -threshold;
t=0; %threshold for pitch
f = 1; %initialization od index
l=(length(x)/length(pd)); %scaling of the index according to the requirement
trade_off = 600; %maximum number of samples allowed between actual and
candidate WB
for i=3:len-3

    if ( e(i-2)==t && e(i-1)==t &&  e(i)~=t &&  e(i+1)~=t && (i-cut_off)>=
threshold )
%        figure(3)
%        subplot(2,1,1)
       cut_off = i;
       index_candidate(f) = ceil(i*l);
         f = f+1;
%       line([ceil(i*l) ceil(i*l)],[0 -0.2],'color','r','LineWidth',3)
    elseif ( e(i-2)~=t && e(i-1)~=t && e(i)==t && e(i+1)==t && (i-cut_off)>=
threshold)
%             figure(3)
%             subplot(2,1,1)
            cut_off = i;
            index_candidate(f) = ceil(i*l);
            f = f+1;
%       line([ceil(i*l) ceil(i*l)],[0 -0.2],'color','r','LineWidth',3)
 elseif ( e(i-2)<t && e(i-1)<t && e(i)>t && e(i+1)>t && (i-cut_off)>=
threshold)
%             figure(3)
%             subplot(2,1,1)
            cut_off = i;
            index_candidate(f) = ceil(i*l);
            f = f+1;
%       line([ceil(i*l) ceil(i*l)],[0 -0.2],'color','r','LineWidth',3)
    end


end
%error calculation

n= length(index_candidate);
ip = index_candidate;
h = 0;

for i= 1:lf
    vector(1:n)= start_sample(i);
```

```
    difference = index_candidate-vector;
    difference = abs(difference);
    [c,j] = min(difference);
   if (c < trade_off && start_sample(i)~=0)
    h = h+1;
    end
end

hits_percentage_pitch = h/n*100 ;%#ok<*NOPTS>
act = length(find(start_sample)); %number of actual word boundaries.
l_pd = length(pd);

fprintf('pitch frames = %4d hits = %4d total cadidates = %4d \n ',l_pd,h,n)
xlswrite('thesis_results', [l_pd,h,n],'pitch','C19:E19')
```

SWIPE code:

```
function [p,t,s] = swipep(x,fs,plim,dt,dlog2p,dERBs,woverlap,sTHR)
% SWIPEP Pitch estimation using SWIPE'.
%    P = SWIPEP(X,Fs,[PMIN PMAX],DT,DLOG2P,DERBS,STHR) estimates the pitch
%    of the vector signal X every DT seconds. The sampling frequency of
%    the signal is Fs (in Hertz). The spectrum is computed using a Hann
%    window with an overlap WOVERLAP between 0 and 1. The spectrum is
%    sampled uniformly in the ERB scale with a step size of DERBS ERBs. The
%    pitch is searched within the range [PMIN PMAX] (in Hertz) with samples
%    distributed every DLOG2P units on a base-2 logarithmic scale of Hertz.
%    The pitch is fine-tuned using parabolic interpolation with a resolution
%    of 1 cent. Pitch estimates with a strength lower than STHR are treated
%    as undefined.
%
%    [P,T,S] = SWIPEP(X,Fs,[PMIN PMAX],DT,DLOG2P,DERBS,WOVERLAP,STHR)
%    returns the times T at which the pitch was estimated and the pitch
%    strength S of every pitch estimate.
%
%    P = SWIPEP(X,Fs) estimates the pitch using the default settings PMIN =
%    30 Hz, PMAX = 5000 Hz, DT = 0.001 s, DLOG2P = 1/48 (48 steps per
%    octave), DERBS = 0.1 ERBs, WOVERLAP = 0.5, and STHR = -Inf.
%
%    P = SWIPEP(X,Fs,...,[],...) uses the default setting for the parameter
%    replaced with the placeholder [].
%
%    REMARKS: (1) For better results, make DLOG2P and DERBS as small as
%    possible and WOVERLAP as large as possible. However, take into account
%    that the computational complexity of the algorithm is inversely
%    proportional to DLOG2P, DERBS and 1-WOVERLAP, and that the  default
%    values have been found empirically to produce good results. Consider
%    also that the computational complexity is directly proportional to the
%    number of octaves in the pitch search range, and therefore , it is
%    recommendable to restrict the search range to the expected range of
```

```
%    pitch, if any. (2) This code implements SWIPE', which uses only the
%    first and prime harmonics of the signal. To convert it into SWIPE,
%    which uses all the harmonics of the signal, replace the word
%    PRIMES with a colon (it is located almost at the end of the code).
%    However, this may not be recommendable since SWIPE' is reported to
%    produce on average better results than SWIPE (Camacho and Harris,
%    2008).
%
%    EXAMPLE: Estimate the pitch of the signal X every 10 ms within the
%    range 75-500 Hz using the default resolution (i.e., 48 steps per
%    octave), sampling the spectrum every 1/20th of ERB, using a window
%    overlap factor of 50%, and discarding samples with pitch strength
%    lower than 0.2. Plot the pitch trace.
%       [x,Fs] = wavread(filename);
%       [p,t,s] = swipep(x,Fs,[75 500],0.01,[],1/20,0.5,0.2);
%       plot(1000*t,p)
%       xlabel('Time (ms)')
%       ylabel('Pitch (Hz)')
%
%    REFERENCES: Camacho, A., Harris, J.G, (2008) "A sawtooth waveform
%    inspired pitch estimator for speech and music," J. Acoust. Soc. Am.
%    124, 1638-1652.
if ~ exist( 'plim', 'var' ) || isempty(plim), plim = [30 5000]; end
if ~ exist( 'dt', 'var' ) || isempty(dt), dt = 0.001; end
if ~ exist( 'dlog2p', 'var' ) || isempty(dlog2p), dlog2p = 1/48; end
if ~ exist( 'dERBs', 'var' ) || isempty(dERBs), dERBs = 0.1; end
if ~ exist( 'woverlap', 'var' ) || isempty(woverlap)
    woverlap = 0.5;
elseif woverlap>1 || woverlap<0
    error('Window overlap must be between 0 and 1.')
end
if ~ exist( 'sTHR', 'var' ) || isempty(sTHR), sTHR = -Inf; end
t = [ 0: dt: length(x)/fs ]'; % Times
% Define pitch candidates
log2pc = [ log2(plim(1)): dlog2p: log2(plim(2)) ]';
pc = 2 .^ log2pc;
S = zeros( length(pc), length(t) ); % Pitch strength matrix
% Determine P2-WSs
logWs = round( log2( 8*fs ./ plim ) );
ws = 2.^[ logWs(1): -1: logWs(2) ]; % P2-WSs
pO = 8 * fs ./ ws; % Optimal pitches for P2-WSs
% Determine window sizes used by each pitch candidate
d = 1 + log2pc - log2( 8*fs./ws(1) );
% Create ERB-scale uniformly-spaced frequencies (in Hertz)
fERBs = erbs2hz([ hz2erbs(min(pc)/4): dERBs: hz2erbs(fs/2) ]');
for i = 1 : length(ws)
    dn = max( 1, round( 8*(1-woverlap) * fs / pO(i) ) ); % Hop size
    % Zero pad signal
    xzp = [ zeros( ws(i)/2, 1 ); x(:); zeros( dn + ws(i)/2, 1 ) ];
    % Compute spectrum
    w = hanning( ws(i) ); % Hann window
    o = max( 0, round( ws(i) - dn ) ); % Window overlap
    [ X, f, ti ] = specgram( xzp, ws(i), fs, w, o );
    % Select candidates that use this window size
    if length(ws) == 1
        j=[(pc)]'; k = [];
    elseif i == length(ws)
```

```
        j=find(d-i>-1); k=find(d(j)-i<0);
    elseif i==1
        j=find(d-i<1); k=find(d(j)-i>0);
    else
        j=find(abs(d-i)<1); k=1:length(j);
    end
    % Compute loudness at ERBs uniformly-spaced frequencies
    fERBs = fERBs( find( fERBs > pc(j(1))/4, 1, 'first' ) : end );
    L = sqrt( max( 0, interp1( f, abs(X), fERBs, 'spline', 0) ) );
    % Compute pitch strength
    Si = pitchStrengthAllCandidates( fERBs, L, pc(j) );
    % Interpolate pitch strength at desired times
    if size(Si,2) > 1
        warning off MATLAB:interp1:NaNinY
        Si = interp1( ti, Si', t, 'linear', NaN )';
        warning on MATLAB:interp1:NaNinY
    else
        Si = repmat( NaN, length(Si), length(t) );
    end
    % Add pitch strength to combination
    lambda = d( j(k) ) - i;
    mu = ones( size(j) );
    mu(k) = 1 - abs( lambda );
    S(j,:) = S(j,:) + repmat(mu,1,size(Si,2)) .* Si;
end
% Fine tune pitch using parabolic interpolation
p = repmat( NaN, size(S,2), 1 );
s = repmat( NaN, size(S,2), 1 );
for j = 1 : size(S,2)
    [ s(j), i ] = max( S(:,j), [], 1 );
    if s(j) < sTHR, continue, end
    if i == 1 || i == length(pc)
        p(j) = pc(i);
    else
        I = i-1 : i+1;
        tc = 1 ./ pc(I);
        ntc = ( tc/tc(2) - 1 ) * 2*pi;
        c = polyfit( ntc, S(I,j), 2 );
        ftc = 1 ./ 2.^[ log2(pc(I(1))): 1/12/100: log2(pc(I(3))) ];
        nftc = ( ftc/tc(2) - 1 ) * 2*pi;
        [s(j) k] = max( polyval( c, nftc ) );
        p(j) = 2 ^ ( log2(pc(I(1))) + (k-1)/12/100 );
    end
end

function S = pitchStrengthAllCandidates( f, L, pc )
% Create pitch strength matrix
S = zeros( length(pc), size(L,2) );
% Define integration regions
k = ones( 1, length(pc)+1 );
for j = 1 : length(k)-1
    k(j+1) = k(j) - 1 + find( f(k(j):end) > pc(j)/4, 1, 'first' );
end
k = k(2:end);
% Create loudness normalization matrix
N = sqrt( flipud( cumsum( flipud(L.*L) ) ) );
for j = 1 : length(pc)
```

```
    % Normalize loudness
    warning off MATLAB:divideByZero
    NL = L(k(j):end,:) ./ repmat( N(k(j),:), size(L,1)-k(j)+1, 1);
    warning on MATLAB:divideByZero
    % Compute pitch strength
    S(j,:) = pitchStrengthOneCandidate( f(k(j):end), NL, pc(j) );
end

function S = pitchStrengthOneCandidate( f, NL, pc )
n = fix( f(end)/pc - 0.75 ); % Number of harmonics
if n==0, S=NaN; return, end
k = zeros( size(f) ); % Kernel
% Normalize frequency w.r.t. candidate
q = f / pc;
% Create kernel
for i = [ 1 primes(n) ]
    a = abs( q - i );
    % Peak's weigth
    p = a < .25;
    k(p) = cos( 2*pi * q(p) );
    % Valleys' weights
    v = .25 < a & a < .75;
    k(v) = k(v) + cos( 2*pi * q(v) ) / 2;
end
% Apply envelope
k = k .* sqrt( 1./f  );
% K+-normalize kernel
k = k / norm( k(k>0) );
% Compute pitch strength
S = k' * NL;

function erbs = hz2erbs(hz)
erbs = 21.4 * log10( 1 + hz/229 );

function hz = erbs2hz(erbs)
hz = ( 10 .^ (erbs./21.4) - 1 ) * 229;
```

Word boundary detection using Teager Energy:

```
function [it,fs] = teager_wb (x,fs,m,lf,start_sample)


%Teager Energy  calculation
l= 1;
N = max(size(x));
teag(1) = x(1)*x(1);
teag(2:N-1) = x(2:N-1).*x(2:N-1)-x(3:N).*x(1:N-2);

teag(N) = x(N)*x(N);
```

```
e= abs(teag);

% Energy(i) now contains the frame-by-frame energy
threshold = 600;
cut_off = -threshold;
t= m;%threshold of energy
trade_off = 600; %threshold for allowable distance between actual and
candidate WB s in frames.

% axis tight
% figure(2)
% plot(x)
% xlabel('Samples')
% ylabel('Amplitude')
% title('speech file and candidate Wbs using Teager Energy')
% hold on
len = length(e);
% axis tight
energy(len+1)= 0;
f = 1;
for i=3:len-3

    if (e(i-2)<= t && e(i-1)<t && e(i)>=t && e(i+1)>=t && e(i+2)>=t && (i-
cut_off)>= threshold)
%        figure(2)
       cut_off = i;
       index_candidate(f) = i*l;
         f = f+1;
%       line([l*i l*i],[0 -0.2],'color','r','LineWidth',3)
    elseif (e(i-2) >= t && e(i-1)>= t && e(i)< t &&  e(i+1)< t && e(i+2)<=t
&& i-cut_off>=threshold)
%             figure(2)
             cut_off = i;
             index_candidate(f) = i*l;
             f = f+1;
%       line([l*i l*i],[0 -0.2],'color','r','LineWidth',3)
    end


end



% for j = 1:lf
%     figure(2)
%     line([start_sample(j) start_sample(j)],[0
0.2],'color','g','LineWidth',3)
% end

%error calculation

n= length(index_candidate);
it = index_candidate;
%start_sample = start_sample';
h = 0;
```

```
for i= 1:lf
    vector(1:n)= start_sample(i);
    difference = index_candidate-vector;
    difference = abs(difference);
    [c,j] = min(difference);
   if (c < trade_off && start_sample(i)~=0)
    h = h+1;
    end
end

hits_percentage_teager = h/n*100 ;%#ok<*NOPTS>
act = length(find(start_sample)); %number of actual word boundaries.

fprintf('Teager Energy frames = %4d hits = %4d total cadidates = %4d \n
',len,h,n)
xlswrite('thesis_results', [len,h,n],'Teager Energy','C19:E19')
```

Word boundary detection using energy:

```
function [ie,fs] = energy_wb (x,fs,m,lf,start_sample)

%frame processing
frame_len=100;
frame_shift=10;
l = frame_len/frame_shift;
num_frames=fix(length(x)/frame_shift-frame_len/frame_shift);

%Energy calculation
energy = zeros(num_frames,1);
for i=1:num_frames
frame=x(((i-1)*frame_shift+1):(i*frame_shift+frame_len));
energy(i)=sum(frame.*frame);
end
e= energy;

% Energy(i) now contains the frame-by-frame energy
threshold = 90;
cut_off = -threshold;
t= m;%threshold of energy
trade_off = 800; %threshold for allowable distance between actual and
candidate WB s in frames.

% figure(1)
% plot(x)
% xlabel('Samples')
% ylabel('Amplitude')
% title('speech file and Candidate Wb s using Energy')
% hold on
```

```
len = length(e);
% axis tight
energy(len+1)= 0;
f = 1;
for i=3:len-3

    if (e(i-2)<= t && e(i-1)<t && e(i)>=t && e(i+1)>=t && e(i+2)>=t && (i-
cut_off)>= threshold )
%        figure(1)
       cut_off = i;
       index_candidate(f) = i*l;
         f = f+1;
%       line([l*i l*i],[0 -0.2],'color','r','LineWidth',3)
    elseif (e(i-2) >= t && e(i-1)>= t && e(i)< t &&  e(i+1)< t && e(i+2)<=t
&& i-cut_off>=threshold)
%             figure(1)
             cut_off = i;
             index_candidate(f) = i*l;
             f = f+1;
%       line([l*i l*i],[0 -0.2],'color','r','LineWidth',3)
    end


end



% for j = 1:lf
%     figure(1)
%     line([start_sample(j) start_sample(j)],[0
0.2],'color','g','LineWidth',3)
% end

%error calculation

n= length(index_candidate);
ie = index_candidate;
%start_sample = start_sample';
h = 0;
for i= 1:lf
    vector(1:n)= start_sample(i);
    difference = index_candidate-vector;
    difference = abs(difference);
    [c,j] = min(difference);

    if (c < trade_off && start_sample(i)~=0)
    h = h+1;
    end
end

hits_percentage_energy = h/n*100; %#ok<*NOPTS>
act = length(find(start_sample)); %number of actual word boundaries.

fprintf('energy frames = %4d hits = %4d total cadidates = %4d \n
',num_frames,h,n)
```

```
xlswrite('thesis_results', [num_frames,h,n],'energy','C19:E19')
```