12

Wake-Up-Word Speech Recognition

Veton Këpuska[1]
Florida Institute of Technology, ECE Department
Melbourne, Florida 32931
USA

1. Introduction

Speech is considered one of the most natural forms of communications between people (Juang & Rabiner, 2005). Spoken language has the unique property that it is naturally learned as part of human development. However, this learning process presents challenges when applied to digital computing systems.
The goal of Automatic Speech Recognition (ASR) is to address the problem of building a system that maps an acoustic signal into a string of words. The idea of being able to perform speech recognition from any speaker in any environment is still a problem that is far from being solved. However, recent advancements in the field have resulted in ASR systems that are applicable to some of Human Machine Interaction (HMI) tasks. ASR is already being successfully applied in application domains such as telephony (automated caller menus) and monologue transcriptions for a single speaker.
Several motivations for building ASR systems are, presented in order of difficulty, to improve human-computer interaction through spoken language interfaces, to solve difficult problems such as speech-to-speech translation, and to build intelligent systems that can process spoken language as proficiently as humans. Speech as a computer interface has numerous benefits over traditional interfaces using mouse and keyboard: speech is natural for humans, requires no special training, improves multitasking by leaving the hands and eyes free, and is often faster and more efficient to transmit than the information provided using conventional input methods.
In the presented work, in Section 2, the concept of Wake-Up-Word (WUW) is being introduced. In Section 3, the definition of the WUW task is presented. The implementation details and experimental evaluations of the WUW-SR system are depicted in Section 4. The WUW recognition paradigm partially applied to general word recognition is presented in Section 5. A developed data collection system from the DVD's of movies and TV series is presented in Section 6. Concluding remarks and future work is provided in Section 7.

2. Wake-up-Word paradigm

In the recent developments (Këpuska & Klein, 2009) were focused on the **Wake-up-Word (WUW)** Speech Recognition paradigm. WUW has the following unique requirement: D*etect*

[1] vkepuska@fit.edu

a single word or phrase when spoken in an alerting context, while rejecting all other words, phrases, sounds, noises and other acoustic events with virtually 100% accuracy including the same word or phrase of interest spoken in a non-alerting (i.e. referential) context (Këpuska, Carstens, & Wallace, 2006)(Këpuska V. , 2006).

One of the goals of speech recognition is to allow *natural* communication between humans and computers via speech, where *natural* implies similarity to the ways humans interact with each other. A major obstacle to this is the fact that most systems today still rely to some extent on non-speech input, such as pushing buttons or clicking with a mouse. However, much like a human assistant, a natural speech interface must be *continuously listening* and must be robust enough to recover from any communication errors without non-speech intervention. Problem with present SR system is that they solely relay on push-to-talk and non-speech intervention paradigms.

1. Push-to-Talk - Speech recognizers deployed in *continuously listening* mode are continuously monitoring acoustic input and do not necessarily require non-speech activation. This is in contrast to the *push- to- talk* model, in which speech recognition is only activated when the user "pushes a button". Unfortunately, today's *continuously listening* speech recognizers are not reliable enough due to their insufficient accuracy, especially in the area of correct rejection. For example, such systems often respond erratically, even when no speech is present. They sometimes interpret a background noise as speech, and they sometimes incorrectly assume that certain speech is addressed at the speech recognizer when in fact it is targeted elsewhere (context misunderstanding). These problems have traditionally been solved by the *push- to- talk* model: requesting the user to push a button immediately before or during talking or similar prompting paradigms. This action in fact represents an explicit triggering of the recognizer while in all other times the recognizer remains inactive, hence avoiding false triggers.
2. Non-Speech Intervention - Another problem with traditional speech recognizers is that they cannot recover from errors gracefully, and often require non-speech intervention. Any speech-enabled human-machine interface based on natural language relies on carefully crafted dialogues. When the dialogue fails, currently there is no good mechanism to resynchronize the communication, and typically the transaction between human and machine fails by termination. A typical example is a SR system which is in a dictation state, when in fact the human is attempting to use command-and-control to correct previous dictation error. Often the user is forced to intervene by pushing a button or keyboard key to resynchronize the system.
3. Current SR systems that do not deploy the *push- to- talk* paradigm use *implicit context switching*. For example, a system that has the ability to switch from "dictation mode" to "command mode" does so by trying to infer whether the user is uttering a command rather than dictating text. This task is rather difficult to perform with high accuracy, even for humans. The *push- to- talk* model uses *explicit context switching*, meaning that the action of pushing the button (or similar paradigm) explicitly sets the context of the speech recognizer to a specific state.

To achieve the goal of developing a natural speech interface, it is first useful to consider human to human communication. Upon hearing an utterance of speech a human listener must quickly make a decision whether or not the speech is directed to him or her. This decision determines whether the listener will make an effort to "process" and understand the utterance. Humans can make this decision quickly and robustly by utilizing visual, auditory, semantic, and/or additional contextual clues.

Visual clues might be gestures such as waving of hands or other facial expressions. Auditory clues are attention grabbing words or phrases such as the listener's name (e.g. John), interjections such as "hey", "excuse me," and so forth. Additionally, the listener may make use of prosodic information such as pitch and intonation, as well as identification of the speaker's voice.

Semantic and/or contextual clues are inferred from interpretation of the words in the sentence being spoken, visual input, prior experience, and customs dictated by the culture. Humans are very robust in determining when speech is targeted towards them, and should a computer SR system be able to make the same decisions its robustness would increase significantly.

Wake-Up-Word is proposed as a method to *explicitly request* the attention of a computer using a spoken word or phrase (Këpuska V. , Elevator Simulator Screen Perspective, 2009), (Këpuska V. , Elevator Simulator User Perspective, 2009). The WUW must be spoken in the context of requesting attention, i. e.alerting context and should not be recognized in any other context. After successful detection of WUW and its alerting context, the speech recognizer may safely interpret the following utterance as a command. The WUW is analogous to the button in *push to talk*, but the interaction is completely based on speech. Therefore it is proposed to use *explicit context switching* via WUW. Furthermore, this is similar to how context switching occurs in human to human communication as well.

3. WUW definition

WUW technology solves three major problem areas:

1. **Detecting WUW Context –** The WUW system must be able to notify the host system that attention is required in certain circumstances and with high accuracy. Unlike keyword-spotting, see for example (Juang & Rabiner, 2005), in which a certain keyword is recognized and reported during *every* occurrence, WUW dictates these occurrences only be reported during an *alerting context*. This context can be determined using features such as leading and trailing silence, difference in the long term average of speech features, and prosodic information (pitch, intonation, rhythm, etc.). This is still active research area (Këpuska & Chih-Ti, 2010)
2. **Identifying WUW –** After identifying the correct context for a spoken utterance, the WUW paradigm shall be responsible for determining if the utterance contains the pre-defined Wake-up-Word to be used for command (e.g. "Computer") with a high degree of accuracy, e.g., > 99% (Këpuska & Klein, 2009).
3. **Correct Rejection of Non-WUW –** Similar to identification of the WUW, the system shall also be capable of filtering speech tokens that are *not* WUWs with practically 100% accuracy to guarantee 0% false acceptances (Këpuska & Klein, 2009).

4. Wake-Up-Word system

The concepts of WUW have been most recently expanded in (Këpuska & Klein, 2009). Currently, the system is implemented in C++ as well as Objective C, and provides four major components for achieving the goals of WUW for use in a real-time environment.

1. **WUW Front End** – This system component is responsible for extracting *features* from the input audio signal. The current system is capable of extracting Mel-Filtered Cepstral Coefficients (MFCC), Linear Predictive Coding coefficients (LPC), and enhanced MFCC features.
2. **Voice Activity Detector (VAD)** – A large portion of the input audio signal to the system are *non-speech events* such as silence or environmental noise. Filtering this information is critical in order to ensure the system is only listening during speech events of interest. Areas of audio that are determined to be speech-related are then forwarded to the later stages of the WUW system.
3. **WUW Back End** – The Back End performs a complex recognition procedure based on Hidden Markov Models (HMMs). HMMs are continuous densities HMM's.
4. **SVM Classification** - The final system component is responsible for classifying speech signals as In-Vocabulary (INV) or Out-of-Vocabulary (OOV) using Support Vector Machines (SVMs). In the WUW context, the only INV word is the one selected for command and control of the host system. Any other word or sound is classified as OOV.

The following diagram illustrates the top-level workflow of the WUW system:

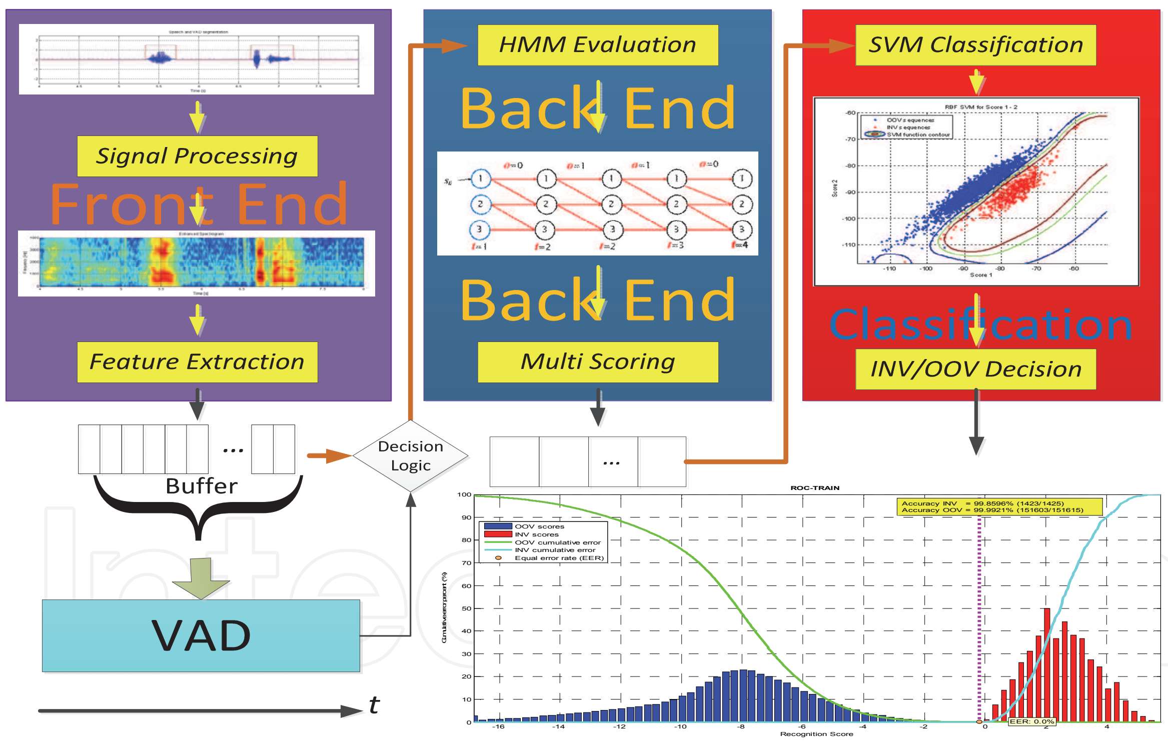


Fig. 1. WUW System Architecture.

4.1 Wake-Up-Word Front End

The front end is responsible for extracting features out of the input signal. Three sets of features are extracted: Mel-Filtered Cepstral Coefficients (MFCC), LPC (Linear Predictive Coding) smoothed MFCCs, and Enhanced MFCCs.

The following image, Figure 2, shows a waveform superimposed with its VAD segmentation, its spectrogram, and its enhanced spectrogram.

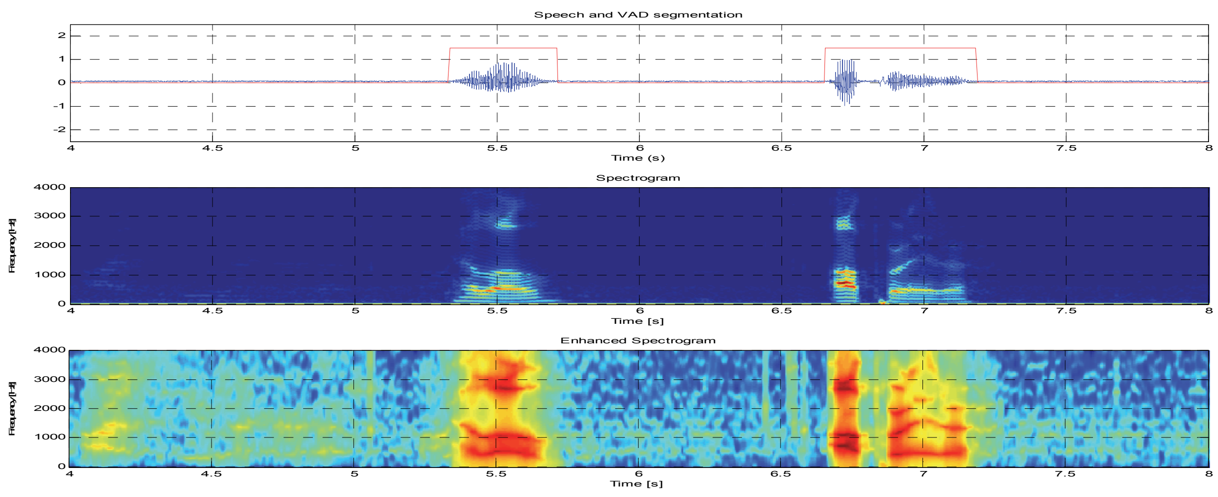


Fig. 2. Speech signal with VAD segmentation, spectrogram, and enhanced spectrogram generated by the Front End module. (Këpuska & Klein, 2009).

The MFCCs are computed using the standard algorithm as presented in the Figure 3:

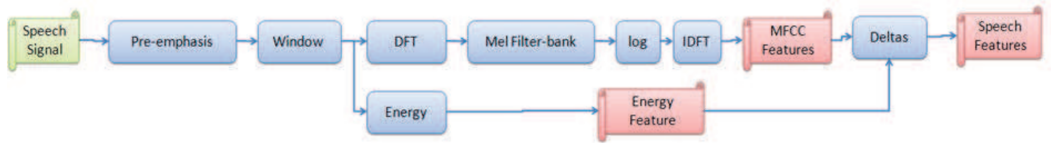


Fig. 3. Feature Extraction Workflow

Pre-emphasis – This stage is used to amplify energy in the high-frequencies of the input speech signal. This allows information in these regions to be more recognizable during HMM model training and recognition.

Windowing – This stage slices the input signal into discrete time segments. This is done by using window *N* milliseconds typically 25 ms wide and at offsets of *M* milliseconds long. A Hamming window is commonly used to prevent edge effects associated with the sharp changes in a Rectangular window. Equation 1 and Figure 5 show the equation for the , typiccally 10 ms or 5 ms Hamming window and its effect when it is applied to a speech signal, respectively:

$$w[n] = \begin{cases} 0.54 - 0.46 \cos \dfrac{2\pi n}{L} & 0 \leq n \leq L-1 \\ 0 & \end{cases} \qquad (1)$$

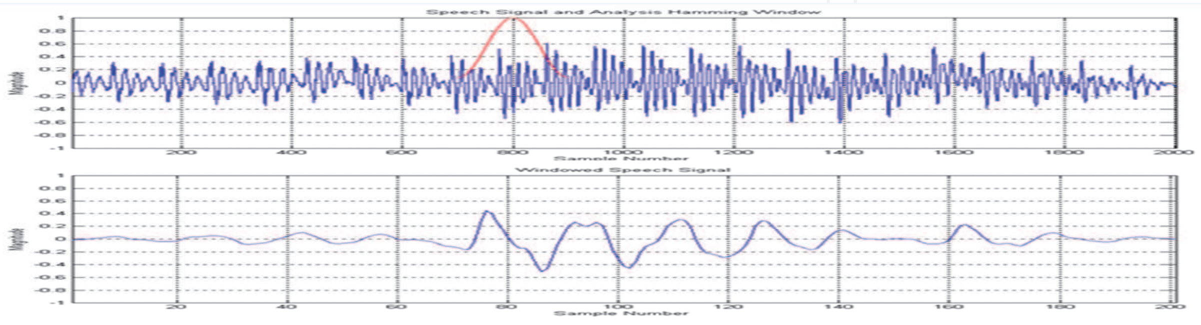


Fig. 4. Original Speech Signal and Windowed Speech Signal

Discrete Fourier Transform – DFT is applied to the windowed speech signal, resulting in the magnitude and phase representation of the signal. The log-magnitude of an example speech signal is depicted in Figure 4.

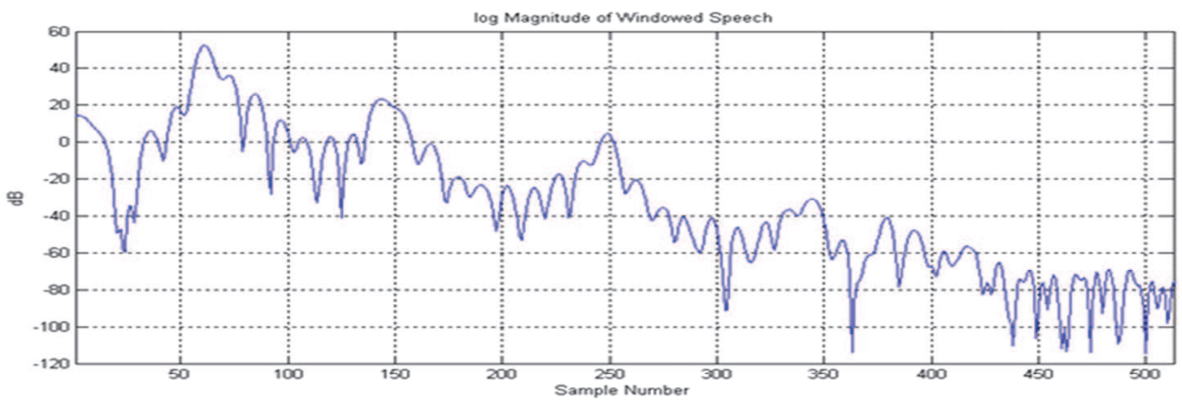


Fig. 5. Log Magnitude of Speech Signal DFT.

Mel Filter Bank – While the resulting spectrum of the DFT contains information in each frequency, human hearing is less sensitive at frequencies above 1000 Hz. This concept also has a direct effect on performance of ASR systems; therefore, the spectrum is warped using a logarithmic **Mel** scale (see Figure 6. below). A Mel frequency can be computed using equation 3. In order to create this effect on the DFT spectrum, a bank of filters is constructed with filters distributed equally below 1000 Hz and spaced logarithmically above 1000 Hz. The Figure 7 displays an example filter bank using triangular filters. The output of filtering the DFT signal by each Mel filter is known as the **Mel spectrum**.

$$mel(f) = 1127 \ln(1 + \frac{f}{700}) \qquad (2)$$

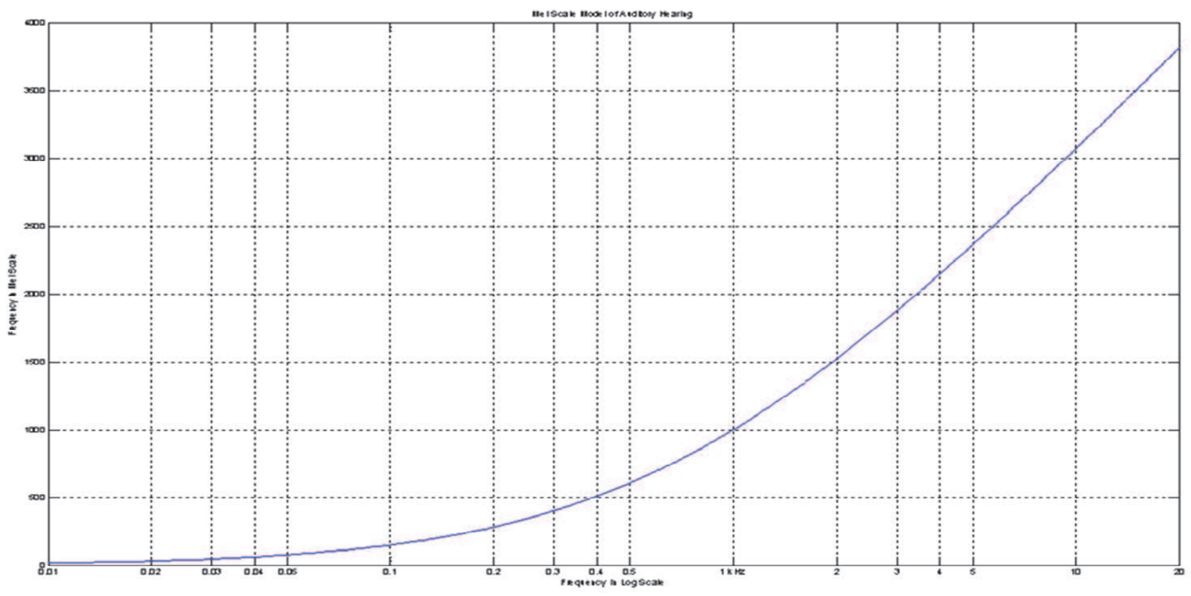


Fig. 6. Mel Scale.

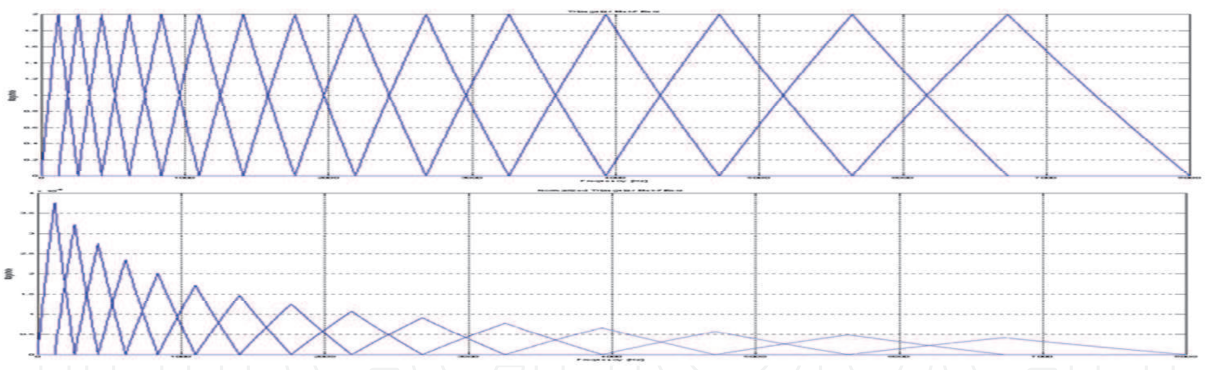


Fig. 7. Mel Filter Bank filters. The bottom figure shows normalized filters.

Inverse DFT – The IDFT of the Mel spectrum is computed, resulting in the **cepstrum**. This representation is valuable because it separates characteristics of the source and vocal tract from the speech waveform. The first 12 values of the resulting cepstrum are recorded.
Additional Features -

i. **Energy Feature** – This step is performed in parallel with the MFCC feature extraction and involves calculating the total energy of the input frame.
ii. **Delta MFCC Features** – In order to capture the changes in speech from frame-to-frame, the first and second derivative of the MFCC coefficients are also calculated and included.

The LPC (Linear Predictive Coding) smoothed MFCCs, and Enhanced MFCCs are described in (Këpuska & Klein, 2009). Those triple features are used by the ***Back End*** to score each with its corresponding HMM model.

4.2 VAD classification

In the first phase, for every input frame VAD decides whether the frame is speech-like or non-speech-like. Several methods have been implemented and tested to solve this problem.
In the first implementation, the decision was made based on three features: log energy difference, spectral difference, and MFCC difference. A threshold was determined empirically for each feature, and the frame was considered speech-like if at least two out of the three features were above the threshold. This was in effect a Decision Tree classifier, and the decision regions consisted of hypercubes in the feature space.
In order to improve the VAD classification accuracy, research has been carried out to determine the ideal features to be used for classification. Hence, Artificial Neural Networks (ANN) and Support Vector Machines (SVM) were tested for automatic classification. One attempt was to take several important features from a stream of consecutive frames and classify them using ANN or SVM. The idea was that the classifier would make a better decision if shown multiple consecutive frames rather than a single frame. The result, although good, was too computationally expensive, and the final implementation still uses information from only a single frame.

4.2.1 First VAD phase – single frame decision

The final implementation uses the same three features as in the first implementation: log energy difference, spectral difference, and MFCC difference; however, classification is performed using a linear SVM. There are several advantages over the original method. First, the classification boundary in the feature space is a hyperplane, which is more robust than

the hypercubes produced by the decision tree method. Second, the thresholds do not have to be picked manually but can be trained automatically (and optimally) using marked input files. Third, the sensitivity can be adjusted in smooth increments using a single parameter, the SVM decision threshold. Recall that the output of a SVM is a single scalar, $u = \boldsymbol{w} \cdot \boldsymbol{x} - b$ (Klein, 2007). Usually the decision threshold is set at $u = 0$, but it can be adjusted in either direction depending on the requirements. Finally, the linear SVM kernel is extremely efficient, because classification of new data requires just a single dot product computation.
The following figures show the training data scattered on two dimensional planes, followed by a 3 dimensional representation which includes the SVM separating plane.

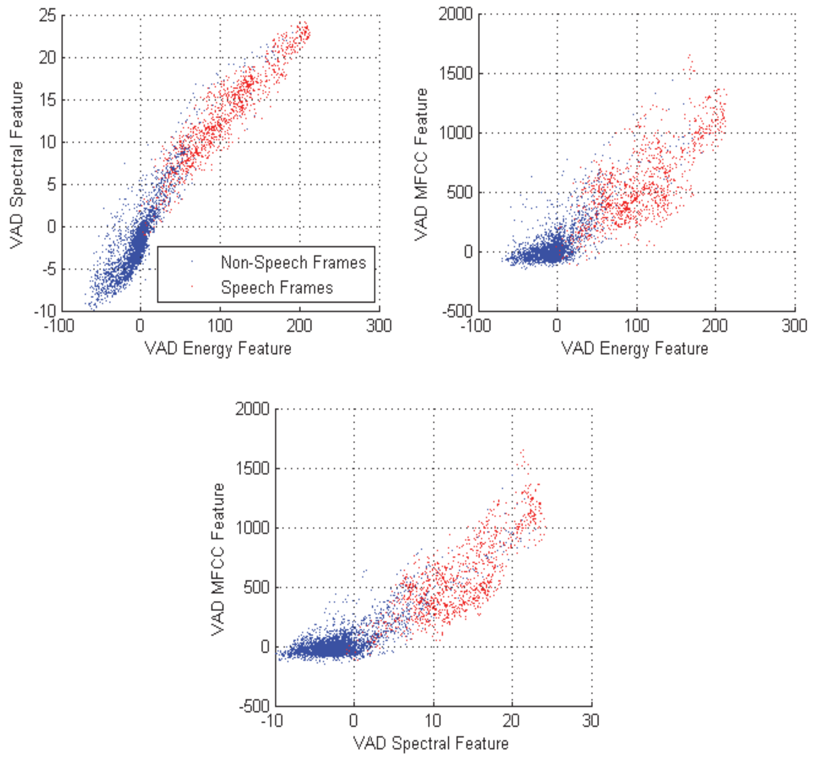


Fig. 8. VAD features in two dimensional space

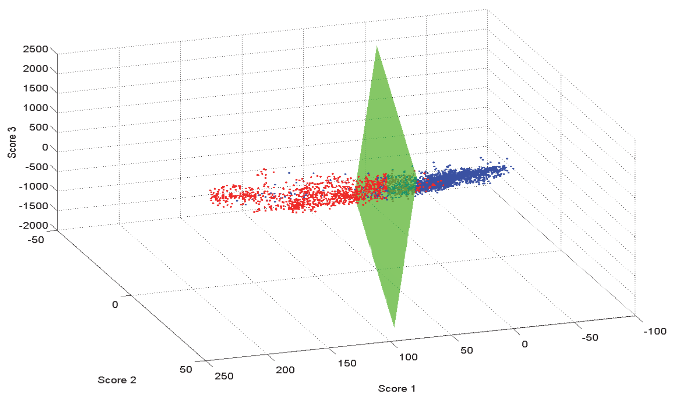


Fig. 9. VAD features with linear SVM

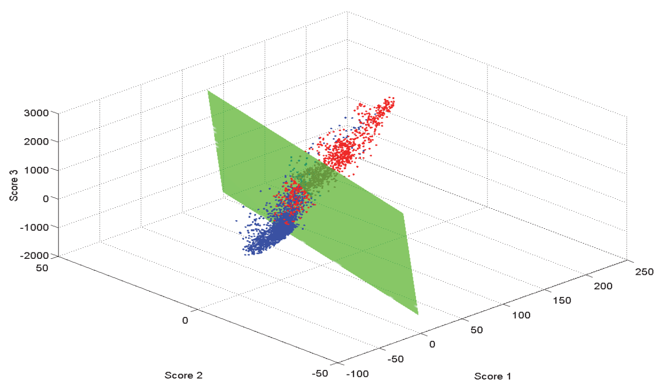


Fig. 10. VAD features with linear SVM (different perspective)

In the figures above, the red points correspond to speech frames while the blue points correspond to non-speech frames, as labeled by a human listener. It can be seen that the linear classifier produces a fairly good separating plane between the two classes, and the plane could be moved in either direction by adjusting the threshold.

4.2.2 Second VAD phase – final decision logic

In the second phase, the VAD keeps track of the number of frames marked as speech and non-speech and makes a final decision. There are four parameters: MIN_VAD_ON_COUNT, MIN_VAD_OFF_COUNT, LEAD_COUNT, and TRAIL_COUNT. The algorithm calls for a number of consecutive frames to be marked as speech in order to set the state to VAD_ON; this number is specified by MIN_VAD_ON_COUNT. It also requires a number of consecutive frames to be marked as non-speech in order to set the state to VAD_OFF; this number is specified by MIN_VAD_OFF_COUNT. Because the classifier can make mistakes at the beginning and the end, the logic also includes a lead-in and a trail-out time. When the minimum number of consecutive speech frames has been observed, VAD does not indicate VAD_ON for the first of those frames. Rather it selects the frame that was observed a number time instances earlier; this number is specified by LEAD_COUNT. Similarly, when the minimum number of non-speech frames has been observed, VAD waits an additional number of frames before changing to VAD_OFF, specified by TRAIL_COUNT.

4.3 Back end - plain HMM scores

The Back End is responsible for scoring observation sequences. The WUW-SR system uses a Hidden Markov Models for acoustic modeling, and as a result the back end consists of a HMM recognizer. Prior to recognition, HMM model(s) must be created and trained for the word or phrase which is selected to be the Wake-Up-Word.
When the VAD state changes from VAD_OFF to VAD_ON, the HMM recognizer resets and prepares for a new observation sequence. As long as the VAD state remains VAD_ON, feature vectors are continuously passed to the HMM recognizer, where they are scored using the novel triple scoring method. If using multiple feature streams, recognition is performed for each stream in parallel. When VAD state changes from VAD_ON to VAD_OFF, multiple scores (e.g., MFCC, LPC and E-MFCC Score) are obtained from the HMM recognizer and are sent to the SVM classifier. SVM produces a classification score which is compared against a threshold to make the final classification decision of INV or OOV.

For the first tests on speech data, a HMM was trained on the word "operator." The training sequences were taken from the CCW17 and WUW-II (Këpuska & Klein, 2009) corpora for a total of 573 sequences from over 200 different speakers. After features were extracted, some of the erroneous VAD segments were manually removed. The INV testing sequences were the same as the training sequences, while the OOV testing sequences included the rest of the CCW17 corpus (3833 utterances, 9 different words, over 200 different speakers). The HMM was a left-to-right model with no skips, 30 states, and 6 mixtures per state, and was trained with two iterations of Baum-Welch.
The score is the result of the Viterbi algorithm over the input sequence. Recall that the Viterbi algorithm finds the state sequence that has the highest probability of being taken while generating the observation sequence. The final score is that probability normalized by the number of input observations, $\boldsymbol{T}$. The Figure 8 below shows the result:
The distributions look Gaussian, but there is significant overlap between them. The equal error rate of 15.5% essentially means that at that threshold, 15.5% of the OOV words would be classified as INV, and 15.5% of the INV words would be classified as OOV. Obviously, no practical applications can be developed based on the performance of this recognizer.

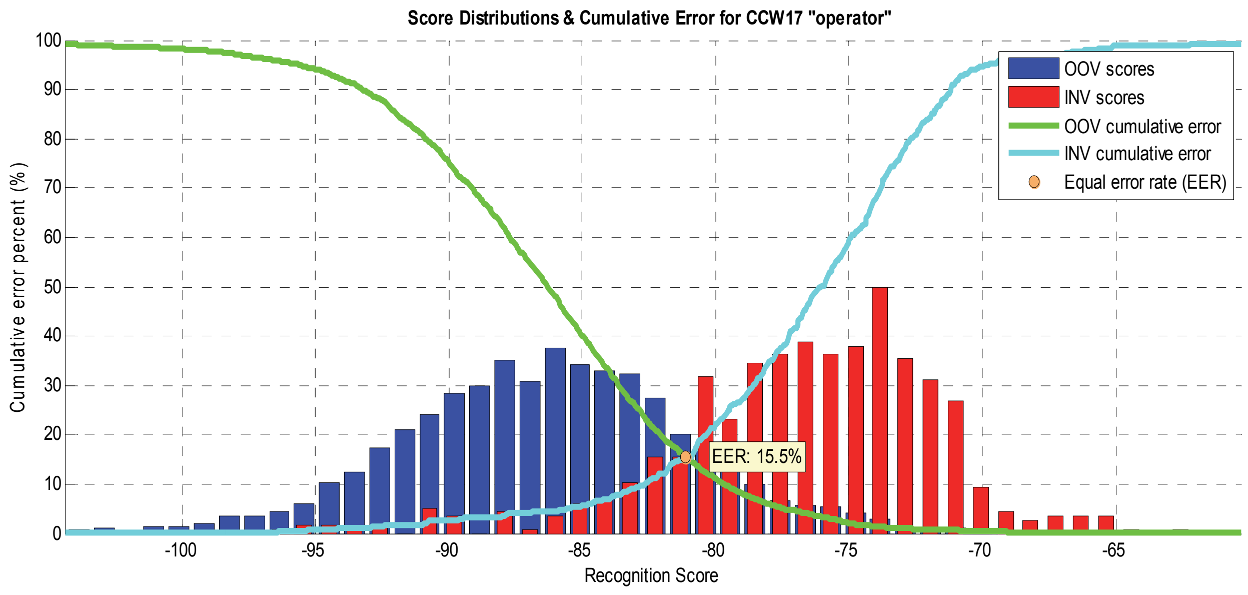


Fig. 11. Score distributions for plain HMM (Këpuska & Klein, 2009)

4.4 SVM classification

After HMM recognition, the algorithm uses two additional scores for any given observation sequence (e.g., MFCC, LPC and e-MFCC). When considering the three scores as features in a three dimensional space, the separation between INV and OOV distributions increases significantly. The next experiment runs recognition on the same data as above, but this time the recognizer uses the triple scoring algorithm to output three scores (Këpuska & Klein, 2009).

4.4.1 Triple scoring method

The figures below show two-dimensional scatter plots of Score 1 vs. Score 2, and Score 1 vs. Score 3 for each observation sequence (e.g., MFCC, LPC and e-MFCC). In addition, a histogram on the horizontal axis shows the distributions of Score 1 independently, and a similar histogram on the vertical axis shows the distributions of Score 2 and Score 3 independently. The histograms are hollowed out so that the overlap between distributions can be seen clearly. The distribution for Score 1 is exactly the same as in the previous section, as the data and model haven't changed. Any individual score does not produce a good separation between classes, and in fact the Score 2 distributions have almost complete overlap. However, the two dimensional separation in either case is remarkable. When all three scores are considered in a three dimensional space, their separation is even better than either two dimensions as depicted in Figure12 and Figure 13.

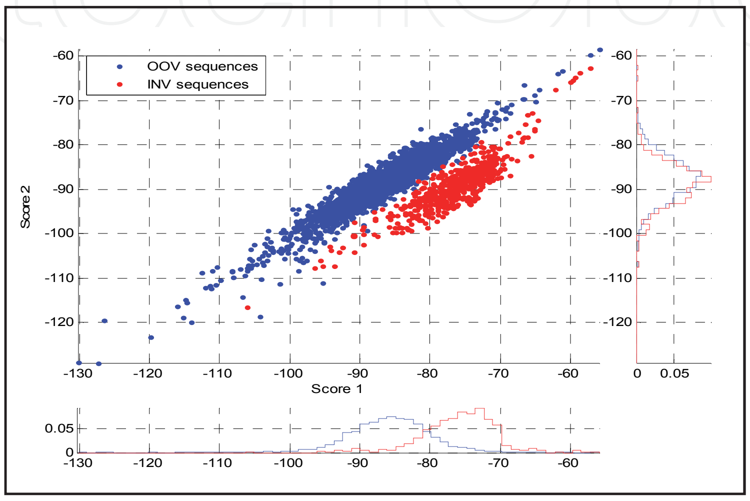


Fig. 12. Triple scoring, score 1 vs. score 2 (Këpuska & Klein, 2009)

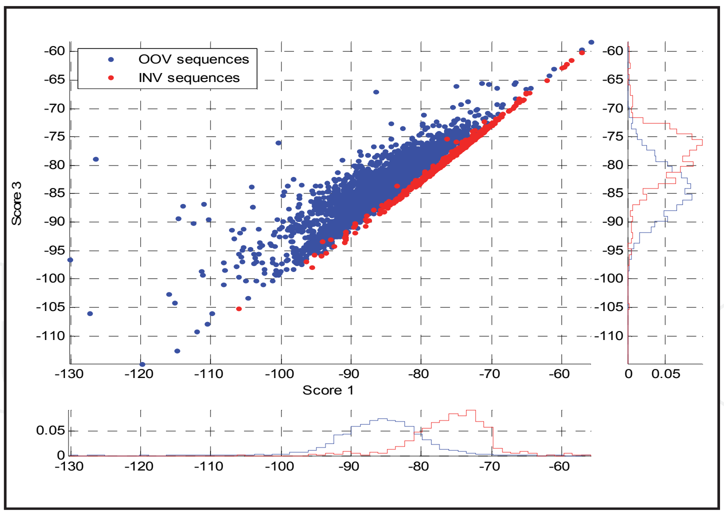


Fig. 13. Triple scoring, score 1 vs. score 3 (Këpuska & Klein, 2009)

In order to automatically classify an input sequence as INV or OOV, the triple score feature space,$\mathcal{R}^3$,can be partitioned by a binary classifier into two regions, $\mathcal{R}_1^3$ and $\mathcal{R}_{-1}^3$. The SVMs have been selected for this task because of the following reasons: they can produce various kinds of decision surfaces including radial basis function, polynomial, and linear; they employ Structural Risk Minimization (SRM) (Burges, 1998) to maximize the margin which has shown empirically to have good generalization performance.

4.4.2 SVM parameters

Two types of SVMs have been considered for this task: linear and RBF. The linear SVM uses a dot product kernel function, $K(x, y) = x \cdot y$, and separates the feature space with a hyperplane. It is very computationally efficient because no matter how many support vectors are found, evaluation requires only a single dot product. Figure 14 above shows that the separation between distributions based on Score 1 and Score 3 is almost linear, so a linear SVM would likely give good results. However, in the Score 1/Score 2 space, the distributions have a curvature, so the linear SVM is unlikely to generalize well for unseen data. The figures below show the decision boundary found by a linear SVM trained on Score 1+2, and Score 1+3, respectively.

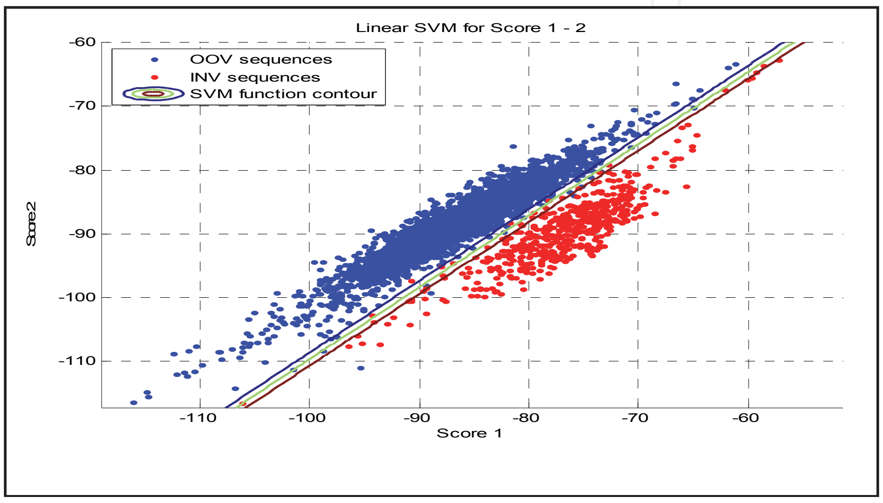


Fig. 15. Linear SVM, scores 1-2 (Këpuska & Klein, 2009)

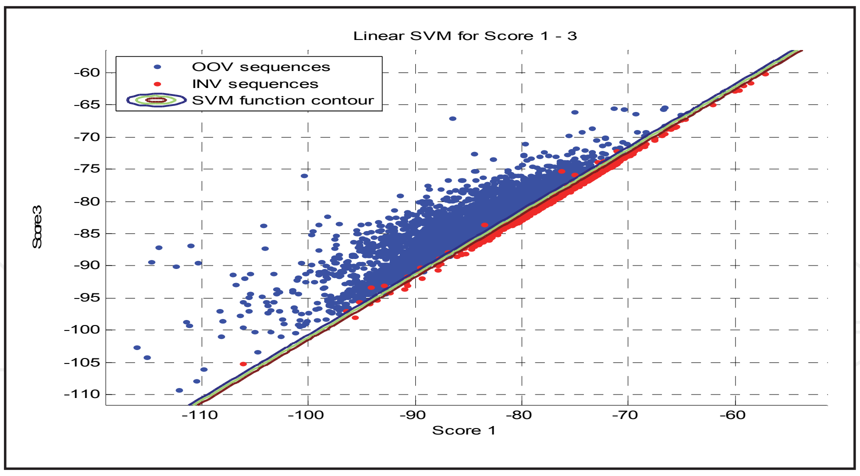


Fig. 16. Linear SVM, scores 1-3 (Këpuska & Klein, 2009)

The line in the center represents the contour of the SVM function at $u = 0$, and outer two lines are drawn at $u = \pm 1$. Using 0 as the threshold, the accuracy of Scores 1 - 2 is 99.7% Correct Rejection (CR) and 98.6% Correct Acceptance (CA), while for Scores 1 – 3 it is 99.5% CR and 95.5% CA. If considering only two features, Scores 1 and 2 seem to have better classification ability. However, combining the three scores produces the plane shown below from two different angles.

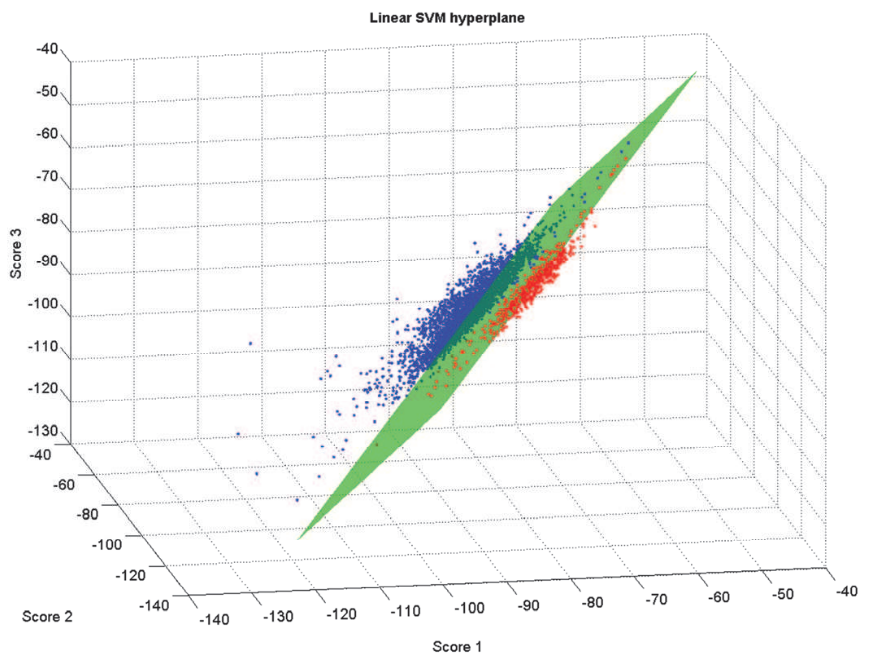


Fig. 17. Linear SVM, 3D view 1 (Këpuska & Klein, 2009)

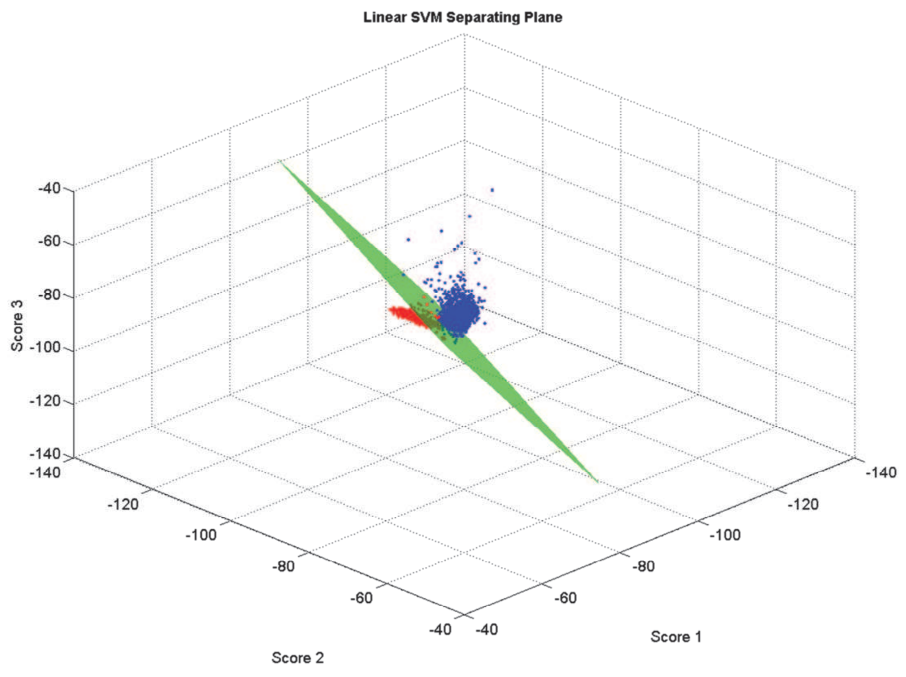


Fig. 18. Linear SVM, 3D view 2 (Këpuska & Klein, 2009)

The plane split the feature space with an accuracy of 99.9% CR and 99.5% CA (just 6 of 4499 total sequences were misclassified). The accuracy was better than any of the 2 dimensional cases, indicating that Score 3 contains additional information not found in Score 2. The classification error rate of the linear SVM is shown below:

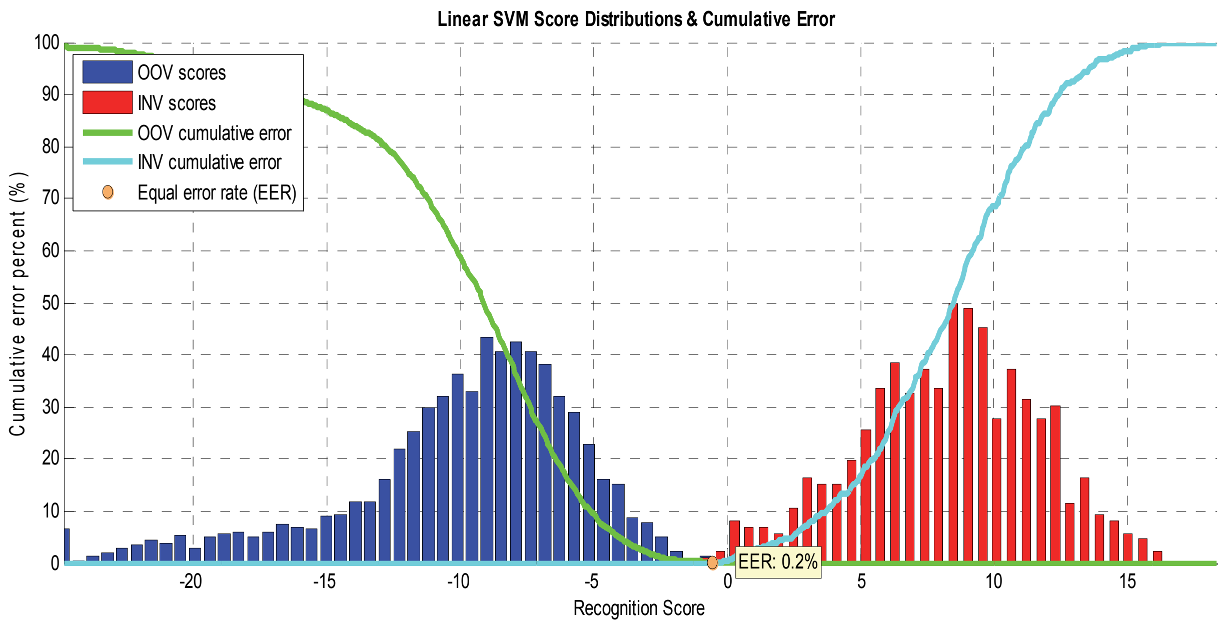


Fig. 19. Score distributions for triple scoring and linear SVM (Këpuska & Klein, 2009)

The conclusion is that using the triple scoring method combined with a linear SVM decreased the equal error rate on this particular data set from 15.5% to 0.2%, or in other words increased accuracy by over 76.5 times (i.e., error rate reduction of 7650%)!

In the next experiment, a Radial Basis Function (RBF) kernel was used for the SVM. The RBF function, $K(x, y) = e^{-\gamma|x-y|^2}$, maps feature vectors into an infinitely dimensional Hilbert space and is able to achieve complete separation between classes in most cases. However, the γ parameter must be chosen carefully in order to avoid overtraining. As there is no way to determine it automatically, a grid search may be used to find a good value. For most experiments $\gamma = 0.008$ gave good results. Shown below are the RBF SVM contours for both two-dimensional cases.

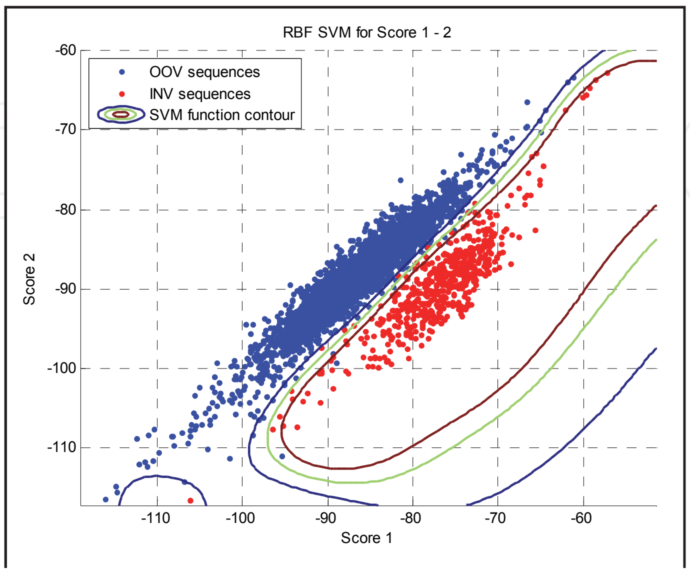


Fig. 20. RBF kernel, scores 1-2 (Këpuska & Klein, 2009)

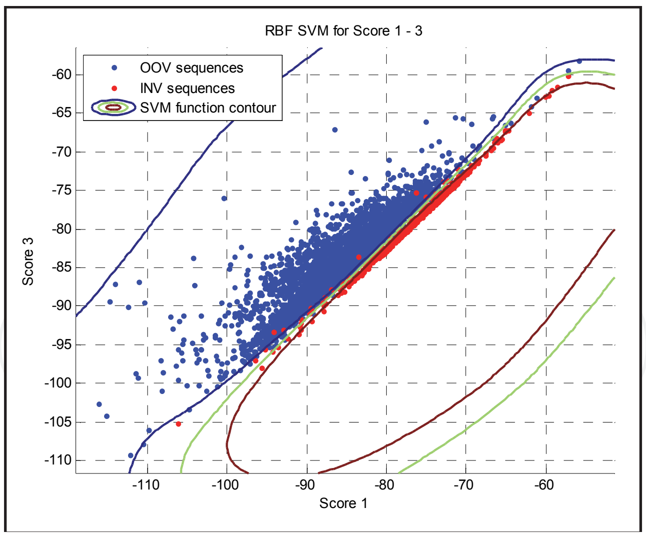


Fig. 21. RBF kernel, scores 1-3 (Këpuska & Klein, 2009)

At the $u = 0$ threshold, the classification accuracy was 99.8% CR, 98.6% CA for Score 1-2, and 99.4% CR, 95.6% CA for Score 1-3. In both cases the RBF kernel formed a closed decision region around the INV points (Recall that the SVM decision function at $u = 0$ is shown by the green line).

Some interesting observations can be made from these plots. First, the INV outlier in the bottom left corner of the first plot caused a region to form around it. SVM function output values inside the region were somewhere between -1 and 0, not high enough to cross into the INV class. However, it is apparent that the RBF kernel is sensitive to outliers, so the γ parameter must be chosen carefully to prevent overtraining. Had the γ parameter been a little bit higher, the SVM function output inside the circular region would have increased beyond 0, and that region would have been considered INV.

Second, the RBF kernel's classification accuracy showed almost no improvement over the linear SVM. However, it is expected that due to the RBF kernel's ability to create arbitrary curved decision surfaces, it will have better generalization performance than the linear SVM's hyperplane.

The figure below shows a RBF kernel SVM trained on all three scores.

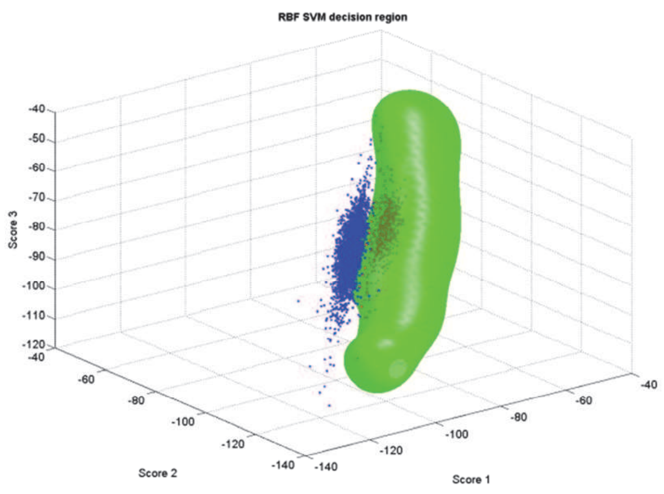


Fig. 22. RBF kernel, 3D view 1 (Këpuska & Klein, 2009)

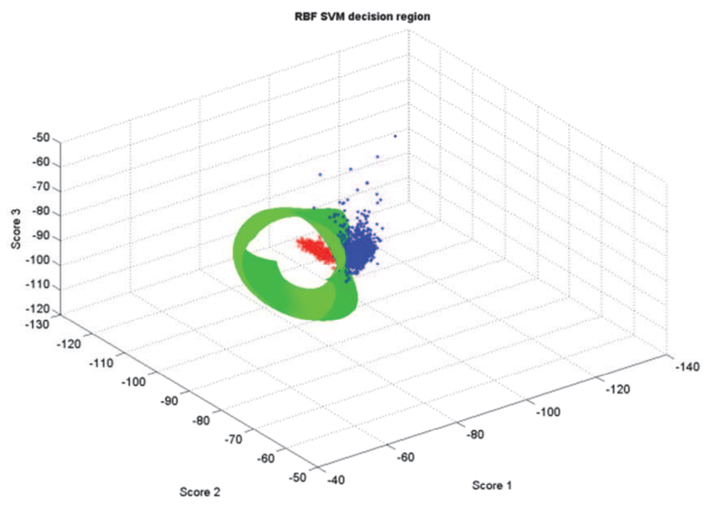


Fig. 23. RBF kernel, 3d view 2 (Këpuska & Klein, 2009)

The RBF kernel created a closed 3-dimensional surface around the INV points and had a classification accuracy of 99.95% CR, 99.30% CA. If considering $u = 0$ as the threshold, the triple score SVM with RBF kernel function shows only little improvement over the linear SVM for this data set. However, as shown below, the SVM score distributions are significantly more separated, and the equal error rate is lower than the linear SVM; from 0.2% to 0.1%.

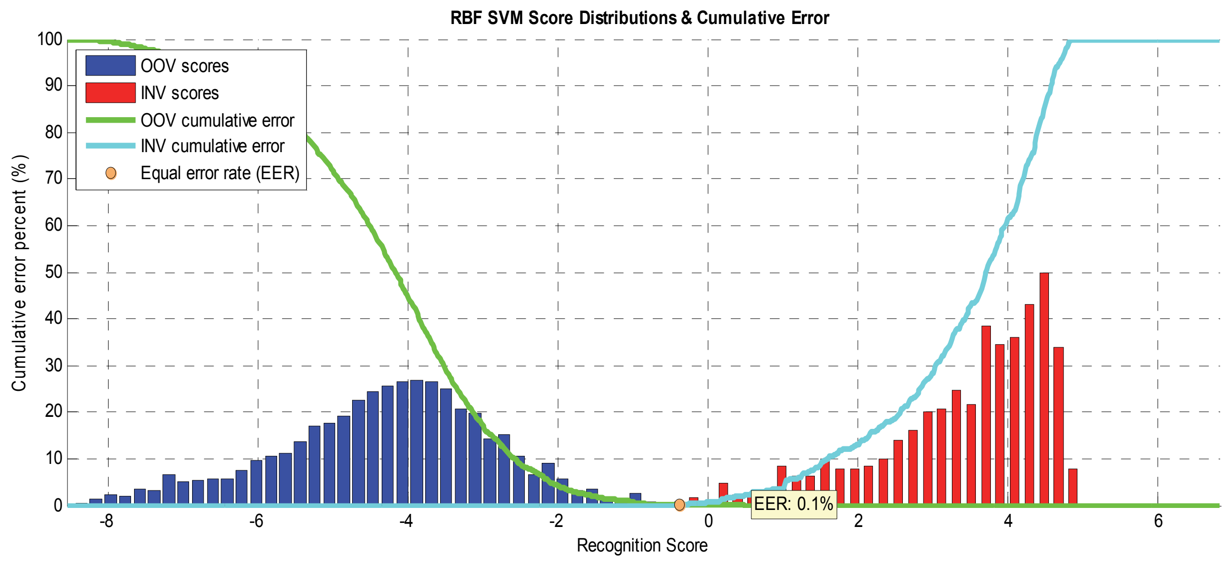


Fig. 24. Score distributions for triple scoring and RBF kernel (Këpuska & Klein, 2009).

Final results when combining all three features using all the available data for testing and training (Callhome, Phonebook, WUW and WUWII Corpora, CCW17) provides a clear superiority of the presented method (Figure 25). In this test the INV accuracy of only two **(2)** errors out of **1425** operator utterances or **99.8596%** and OOV accuracy of twelve **(12)** errors on **151615** tokens or **99.9921%**.

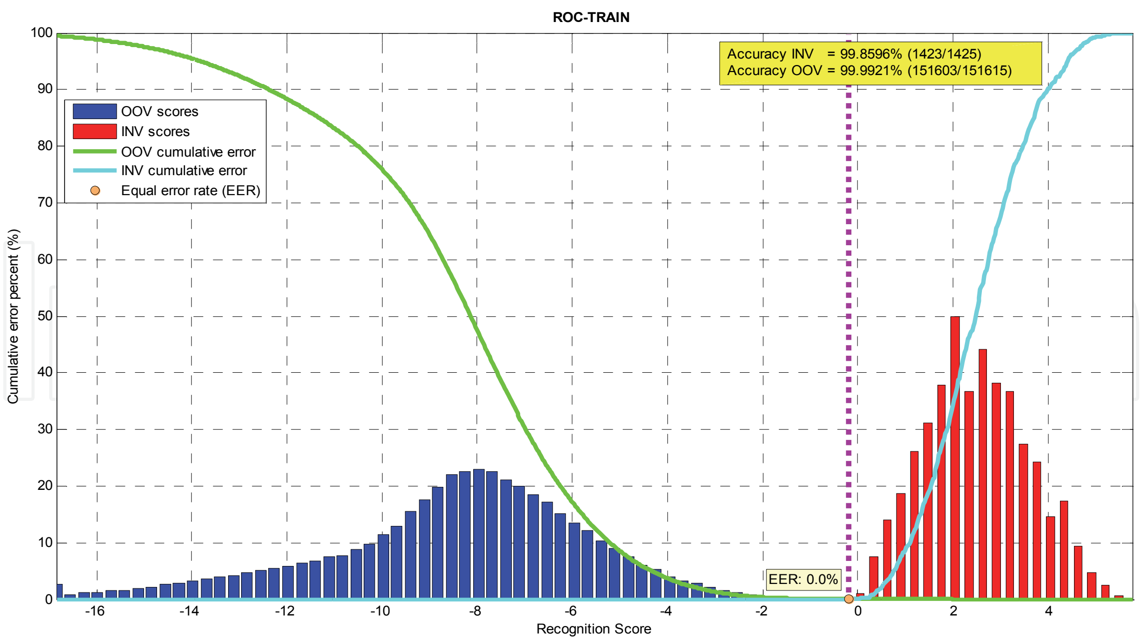


Fig. 25. Overall recognition rate of WUW-SR utilizing Callhome, Phonebook, WUW, WUWII and CCW17 corpora.

The chart above demonstrates WUW-SR's improvement over current state of the art recognizers. WUW-SR with three feature streams was several orders of magnitude superior to the baseline recognizers in INV recognition and particularly in OOV rejection. Of the two OOV corpora, Phonebook was significantly more difficult to recognize for all three systems due to the fact that every utterance was a single isolated word. Isolated words have similar durations to the WUW and differ just in pronunciation, putting to the test the raw recognition power of ASR system. HTK had 8793 false acceptance errors on Phonebook and the commercial SR system had 5801 errors.

WUW-SR demonstrated its OOV rejection capabilities by committing just total of **12** false acceptance errors on all the corpora (**151615 tokens**) used for OOV rejection testing while maintaining high recognition rate by committing only **2** false rejection errors for **1425** INV words. This result is not being biased to optimize against neither false rejection nor false acceptance. If biasing is necessary it can be easily accomplished by shifting the threshold factor of SVM from zero toward -1 to reduce false rejection or toward +1 to reduce false acceptance.

5. Scoring experiments using the TIMIT corpus and the HTK

In order to show the versatility of the approach taken in the whole word HMM scoring with WUW-SR paradigm the next set of experiments applied a phonetic modeling approach. The experiments rely on the **Texas Instruments and Massachusetts Institute of Technology (TIMIT)** corpus. TIMIT is a standard data set that is designed to provide speech data for acoustic-phonetic studies and for the development and evaluation of ASR systems (John S. Garofolo, 1993). TIMIT contains recordings of 630 speakers in 8 dialects of U.S. English. Each speaker is assigned 10 sentences to read that are carefully designed to contain a wide range of phonetic variability. Each utterance is recorded as a 16-bit waveform file sampled at 16 KHz. The entire data set is split into two portions: *TRAIN* to be used to generate an SR baseline, and *TEST* that should be **unseen** by the experiment until the final evaluation.

5.1 Data selection

TIMIT contains time-aligned transcriptions on both the whole word and phonetic level. Each transcription is hand-verified for proper time-boundary markings.[2] TIMIT also includes a phonetic dictionary containing the transcriptions for all words. The combined, unique word count of each utterance in TIMIT is ~**6220 words**. However, valid training and evaluation can only be performed on individual words that have a reasonable amount of occurrences within the *TEST* portion of the data set.

It is important to note that the selected test set contains the *most common* pronunciation of the word. The difference in phoneme choices for the alternate pronunciation is not considered for the experiments.

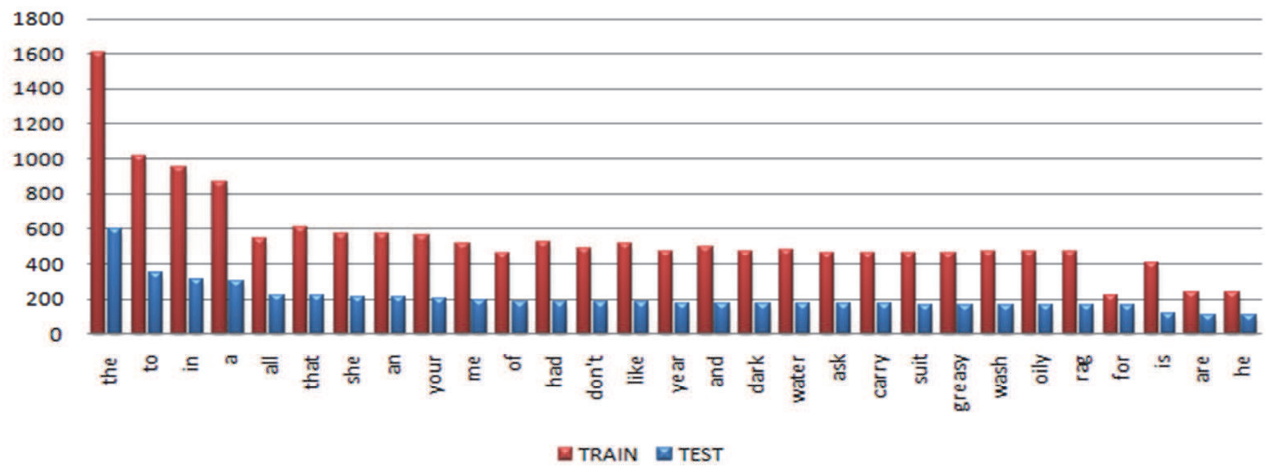


Fig. 26. TIMIT Word Occurrences

5.2 Baseline score

Using the two-score method and a properly trained SVM model, the Total Error Rate is reduced from 2.45% (standard scoring using HTK) to 0.97%; a recognition rate that is **2.55** times better**.**

The developed single-word recognition system has the following techniques that will be evaluated against the 25-word TIMIT *TEST* subset. Each technique will be assigned a code that will be referred to in the presented results.

1. (**Score 1**) Standard Acoustic Score Classification (Code: "Score 1")
2. (**Score 1 + Score 2**) Classification With Two-Class SVM (Code: "CSVM No Dur.")
3. (**Score 1 + Score 2 + Duration**) Classification With Two-Class SVM (Code: "CSVM")
4. (**Score 1 + Score 2 + Duration**) Classification With One-Class SVM (Code: "OSVM")

All acoustic scores will be generated using **16-Mixture Monophone HMMs**. The Total Error Rate metric will be the primary criterion of performance for each method described above. The Relative Error Rate Reduction (RERR) and Error Rate Ratio (ERRR) will be calculated and used to compare performance between two methods. They are computed as:

[2] While useful, the accuracy of many of the time boundary markings in TIMIT is questionable.

$$\text{RERR} = \frac{B - N}{N} * 100\%;\ \text{ERRR} = \frac{B}{N} \tag{3}$$

where B is the baseline Total Error Rate and N is the new Total Error Rate.

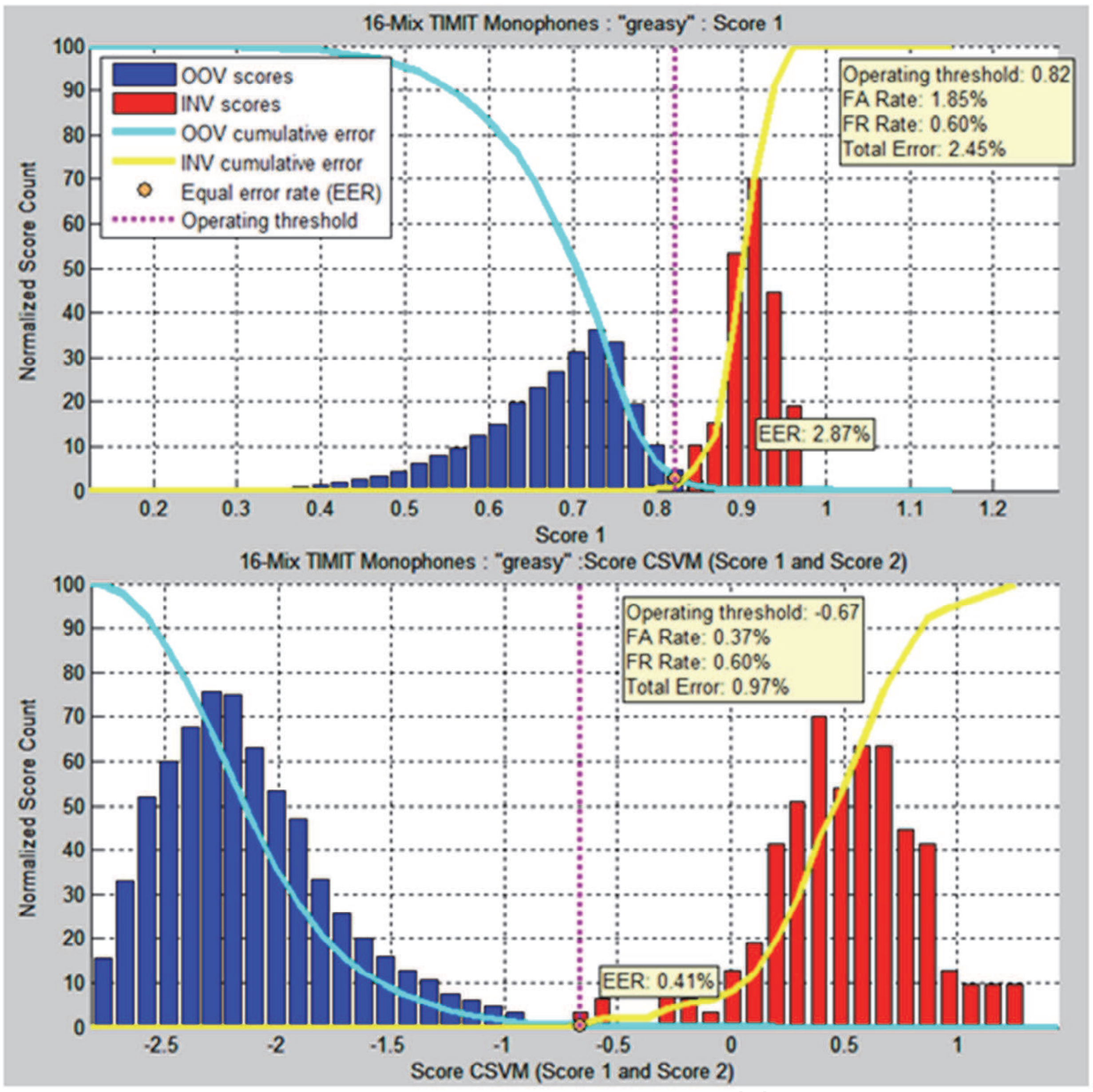


Fig. 27. SVM Classifier Scores for TIMIT word "greasy". Top viewgraph uses standard score (Score 1), the second viewgraph uses Score1 and Score2.

5.3 Summary results

The entire TIMIT test set results are shown in Figure 28 and Figure 29. The word list is sorted by increasing Total Error Rates for the usual, standard "Score 1", classifier and broken into two groups for clarity. The Table 1 and Table 2 summarize the RERR gains for each method. Table 2 also shows the average RERR gain across *all* words considered in the TIMIT test set.

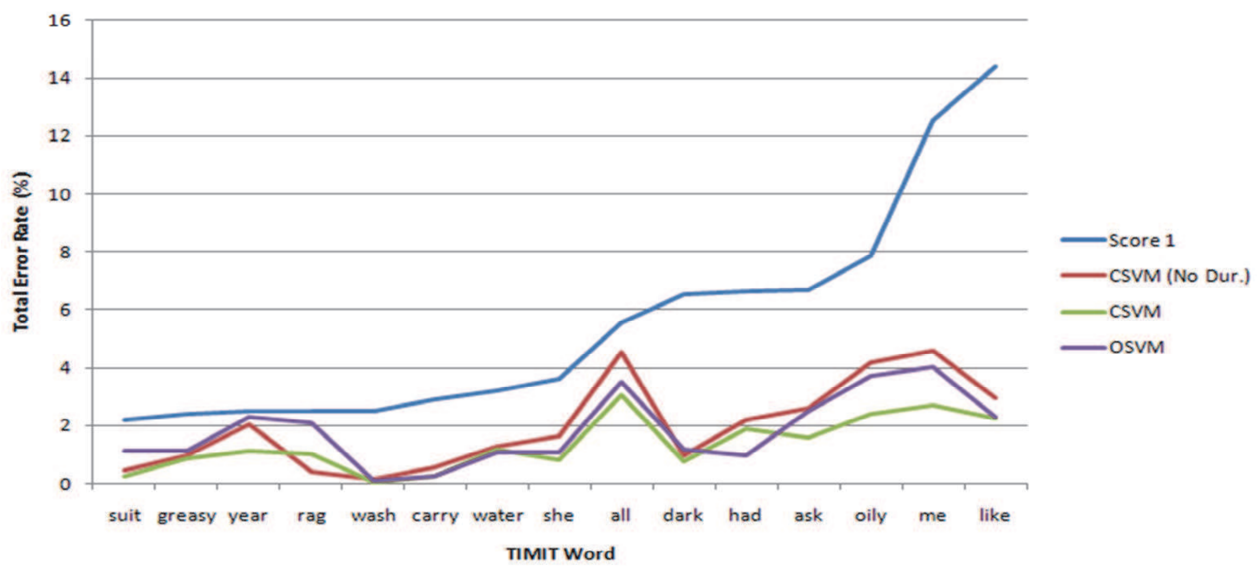


Fig. 28. Error Rates – for TIMIT Test Set (Part 1)

Word	RERR (%) CSVMND	ERRR CSVMND	RERR (%) CSVM	ERRR CSVM	RERR (%) OSVM	ERRR OSVM
suit	391%	4.91	723%	8.23	90%	1.90
greasy	146%	2.46	172%	2.72	117%	2.17
year	23%	1.23	126%	2.26	8%	1.08
rag	518%	6.18	147%	2.47	19%	1.19
wash	1432%	15.32	4303%	44.03	2249%	23.49
carry	407%	5.07	1152%	12.52	1152%	12.52
water	156%	2.56	180%	2.80	197%	2.97
she	124%	2.24	344%	4.44	245%	3.45
all	22%	1.22	82%	1.82	59%	1.59
dark	590%	6.90	737%	8.37	465%	5.65
had	205%	3.05	253%	3.53	575%	6.75
ask	158%	2.58	324%	4.24	169%	2.69
oily	89%	1.89	228%	3.28	111%	2.11
me	175%	2.75	368%	4.68	211%	3.11
Like	389%	4.89	538%	6.38	522%	6.22

Table 1. Summary of RERR Gains for the WUW Recognizer – TIMIT Test Set (Part 1)

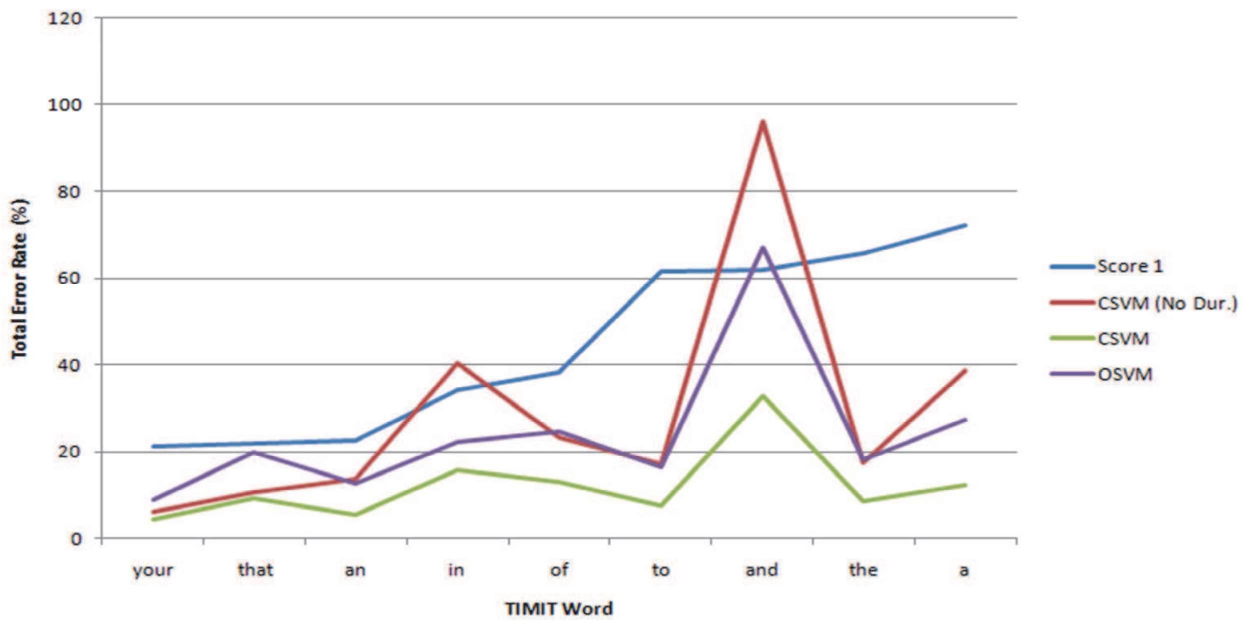


Fig. 29. Error Rates – for TIMIT Test Set (Part 2)

Word	RERR (%) CSVMND	ERRR CSVMND	RERR (%) CSVM	ERRR CSVM	RERR (%) OSVM	ERRR OSVM
your	251%	3.51	360%	4.60	137%	2.37
that	106%	2.06	133%	2.33	10%	1.10
an	64%	1.64	301%	4.01	79%	1.79
in	-15%	0.85	116%	2.16	53%	1.53
of	65%	1.65	198%	2.98	55%	1.55
to	261%	3.61	725%	8.25	278%	3.78
and	-36%	0.64	88%	1.88	-8%	0.92
the	275%	3.75	673%	7.73	258%	3.58
Average	**245%**	**3.45**	**532%**	**6.32**	**301%**	**4.00**

Table 2. Table Summary of RERR Gains for *the WUW* Recognizer – TIMIT Test Set (Part 2)

The final **r**esults in Table 6 show an average Relative Error Rate Reduction of **532%** using Two-Class SVM Scoring with the Duration feature. This leads to a word recognition system capable of performing with an overall average Total Error Rate of **5.4%** as compared to the baseline of **20.6%**. The highest gain was found using the TIMIT word "wash": the baseline Total Error Rate was **2.51%** and the Two-Class SVM with Duration Total Error Rate was **0.06%**; a RERR of **4303%**!
The results also shows that One-Class SVM is indeed a viable method for significantly reducing recognizer error, with an average RERR of **301%** which still outperforms Two-

Class SVM without the Duration feature. Note that Once-Class SVM is necessary if the OOV data is not available.
The McNemar test for statistical significance, presented in (Gillick & Cox, 1989), was performed to compare results between the Score 1 recognizer and the Two-Class SVM with Duration recognizer. Two tests were performed to characterize the error rates of correct recognition (INV) and correct rejection (OOV). Across all TIMIT test data, on average, both the INV and OOV showed statistical relevance, with INV P= 4.67E-01 and OOV P = 2.25E-04. In other words, the probability of the null hypothesis (that both methods are statistically the same) for an INV test is on average 47% and for OOV is 0.02%. Note that the test for INV data is skewed due to some difficult short words.

6. Data collection system

The idea of corpus generation from DVDs of movies and TV series is inspired from the study of prosodic analysis of Wake-Up-Word technology (Placeholder1) (Këpuska & Shih, 2010) and (Këpuska, Sastraputera, & Shih, Discrimination of Sentinel Word Contexts using Prosodic Features, Submitted 2010). A need for obtaining natural and inexpensive speech corpora for prosodic analysis and in general study for any speech recognition had inspired the research for the data collection system from DVDs of movies and TV series. The C# .NET Framework application Data Collection Toolkit was developed to facilitate this research.

6.1 Components of data collection system

The data collection system consists of five components. The first step of the data collection system begins from selecting a DVD of a movie or TV series. The audio will then be extracted from the video of the DVD. The text from the subtitles will be converted into text transcriptions. The subtitles will also be used for cutting the extracted audio file into small utterances (audio segmentation). The text transcriptions and utterances are then combined into a basic speech corpus.
There are two optional components in the data collection system that can be used with the corpora generated from the DVDs for more specific tasks. First is the language analysis: this component takes a text from a transcription and finds the relationship of each word. This analysis can be used for the study of Wake-Up-Word's context detection. Another component is the force alignment: which can be used for generating time markers of each individual word in the selected utterance. The force alignment can also be used for filtering and/or trimming utterances to ensure that they match with the corresponding text transcriptions. The Figure 30 shows an overview of the data collection system.
The Data Collection Toolkit provides a tool called TIMIT Corpus Browser, which allows a user to load and view TIMIT corpus in a form that is more organized and easier to view TIMIT's utterances as shown in Figure 31. It also provides options to play an audio of any utterance and its individual words using time markers labeled by TIMIT or SAPI Force Alignment (if available). There is also an option to export the information of the TIMIT corpus into a Comma Separated Value (CSV) file to perform further data analysis such as drawing graphs, histograms, or any other numerical computation using Microsoft Excel spreadsheet.

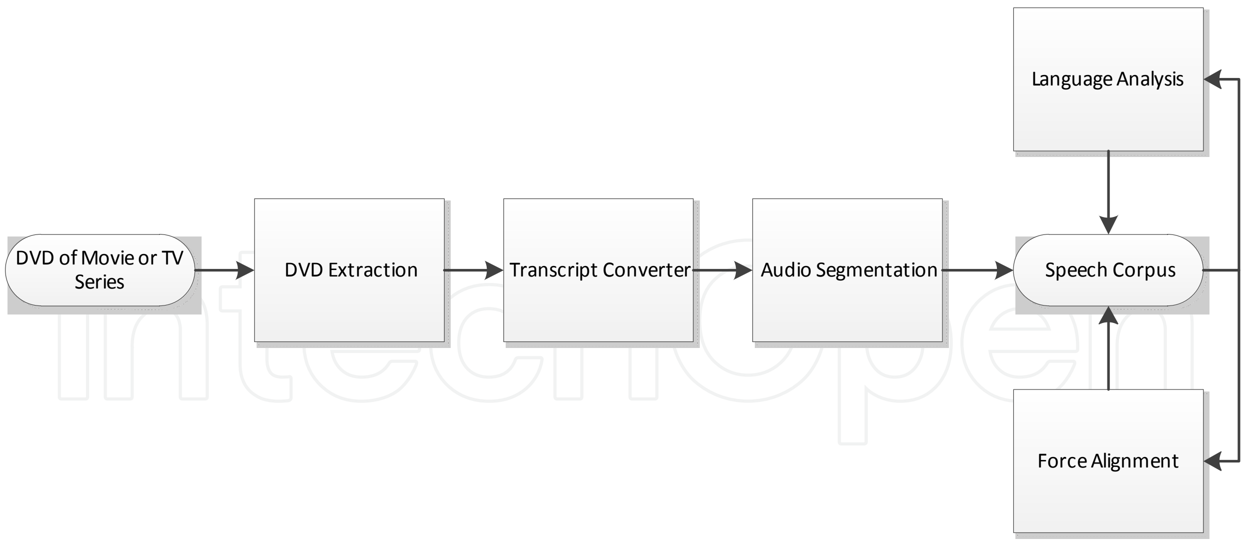
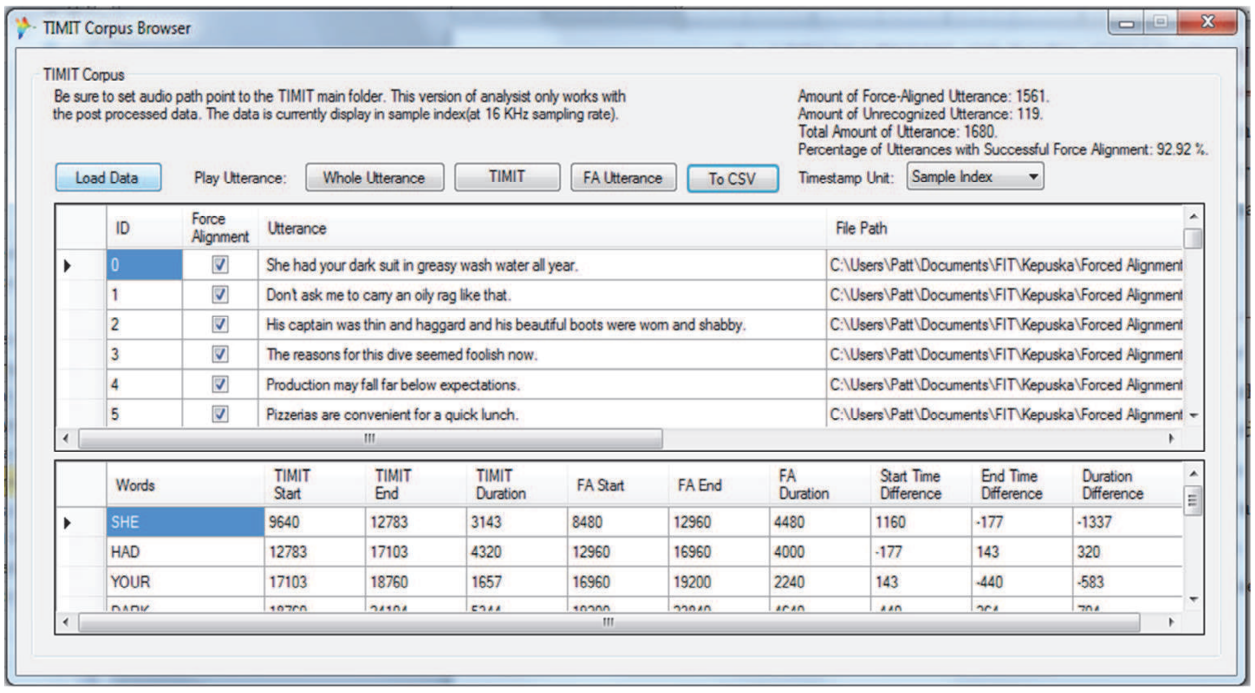


Fig. 30. Overview of Data Collection System



TIMIT Corpus

Be sure to set audio path point to the TIMIT main folder. This version of analysist only works with the post processed data. The data is currently display in sample index(at 16 KHz sampling rate).

Amount of Force-Aligned Utterance: 1561.
Amount of Unrecognized Utterance: 119.
Total Amount of Utterance: 1680.
Percentage of Utterances with Successful Force Alignment: 92.92 %.

ID	Force Alignment	Utterance	File Path
0	✓	She had your dark suit in greasy wash water all year.	C:\Users\Patt\Documents\FIT\Kepuska\Forced Alignment
1	✓	Dont ask me to carry an oily rag like that.	C:\Users\Patt\Documents\FIT\Kepuska\Forced Alignment
2	✓	His captain was thin and haggard and his beautiful boots were worn and shabby.	C:\Users\Patt\Documents\FIT\Kepuska\Forced Alignment
3	✓	The reasons for this dive seemed foolish now.	C:\Users\Patt\Documents\FIT\Kepuska\Forced Alignment
4	✓	Production may fall far below expectations.	C:\Users\Patt\Documents\FIT\Kepuska\Forced Alignment
5	✓	Pizzerias are convenient for a quick lunch.	C:\Users\Patt\Documents\FIT\Kepuska\Forced Alignment

Words	TIMIT Start	TIMIT End	TIMIT Duration	FA Start	FA End	FA Duration	Start Time Difference	End Time Difference	Duration Difference
SHE	9640	12783	3143	8480	12960	4480	1160	-177	-1337
HAD	12783	17103	4320	12960	16960	4000	-177	143	320
YOUR	17103	18760	1657	16960	19200	2240	143	-440	-583

Fig. 31. Screenshot of TIMIT Corpus Browser

6.2 Performance evaluation of the system

To compare the time markers, the TIMIT Corpus Browser tool can export the information into the CSV file. The content of this file is then loaded into Microsoft Excel and used for drawing histograms as shown in Figure 32. Based on these histograms, the most of the frequency difference values of the time markers and word durations are located between -50 to 50 milliseconds and the largest frequency is at 0. Some extreme differences values turned out to be that the time markers of either from TIMIT corpus or SAPI Force Alignment's results have included the silence in them.

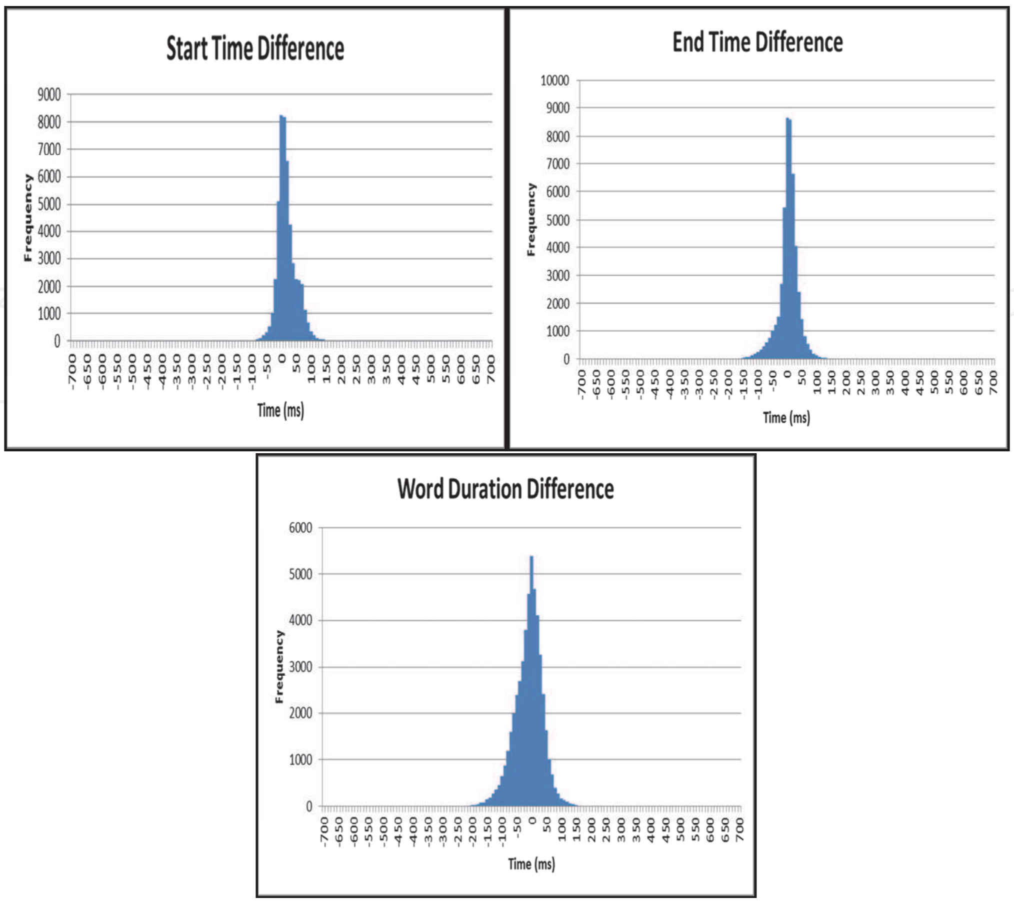


Fig. 32. Start and End Time as well as Word Duration Difference between TIMIT and SAPI Force Alignment

7. Conclusion

The WUW-SR system developed in this work provides for efficient and highly accurate speaker independent recognitions at performance levels not achievable by current state of the art recognizers. Extensive testing demonstrates accuracy improvements superior by several orders of magnitude over the best known academic speech recognition system, HTK, as well as a leading commercial speech recognition system. Specifically, the WUW-SR system correctly detects the WUW with **99.98%** accuracy. It correctly rejects non-WUW with over **99.99%** accuracy. The WUW system makes **12** errors in **151615** words or less than **0.008%**. Assuming speaking rate of **100 words per minute** it would make **0.47** false acceptance errors **per hour**, or **one false acceptance** in **2.1 hours**.

Comparison of WUW performance in detection and recognition performance is **2525%**, or **26** times better than HTK for the same training & testing data, and **2,450%**, or **25** times better than Microsoft SAPI 5.1 recognizer. The out-of-vocabulary rejection performance is over **65,233%,** or **653** times better than HTK, and **5900%** to **42,900%**, or **60** to **430** times better than the Microsoft SAPI 5.1 recognizer.

In order to achieve these levels of accuracy, the following innovations were accomplished:

- Hidden Markov Model triple scoring with Support Vector Machine classification
- Combining multiple speech feature streams (MFCC, LPC-smoothed MFCC, and Enhanced MFCC)

- Improved Voice Activity Detector with Support Vector Machines classification

To show that the presented WUW paradigm can be applied to Large Vocabulary Continuous Speech Recognition (LVCSR), the experiments using HTK framework using TIMIT corpus and partial WUW paradigm (e.g., Score1 and Score2) were conducted. The results show superior performance of the recognition compared to the HTK: an average Relative Error Rate Reduction of **532%** using Two-Class SVM Scoring with the Duration feature was obtained.

Finally, the data collection frame work from the DVD's of movies and TV series was described. It provides a good source of data against expensive data collection systems. The system was evaluated against TIMIT's manually labeled time markers that showed no statistical difference. In the future it is planned to apply the developed data collection system for further data collection that will serve for future development of a LVCSR system.

8. References

AoAMedia.Com. (2009). AoA Audio Extractor.

Burges, C. J. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*.

Cambridge University Engineering Department. (n.d.). *HTK 3.0.* Retrieved 2007, from http://htk.eng.cam.ac.uk

Carnegie Mellon University. (n.d.). Retrieved April 2007, from The CMU Pronouncing Dictionary: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

Chang, C.-C., & Lin, C.-J. (2001). *LIBSVM : A Library for Support Vector Machines.* Retrieved from http://www.csie.ntu.edu.tw/~cjlin/libsvm

CMU Sphinx. (2010). *CMU Sphinx: Open Source Toolkit for Speech Recognition*. Retrieved 8 13, 2010, from http://cmusphinx.sourceforge.net/

Daniels, G. (Director). (2005). *The Office Season 1* [Motion Picture].

Fan, R.-E., Chen, P.-H., & Lin, C.-J. (2005). Working Set Selection Using Second Order Information for Training Support Vector Machines. *Journal of Machine Learning Research 6*, pp. 1889-1918.

Garofolo, John S., et al. (1993). TIMIT Acoustic-Phonetic Continuous Speech Corpus. Philadelphia, Pennsylvania, USA.

Gillick, L., & Cox, S. (1989). Some Statistical Issues in the Comparison of Speech Recognition Algorithms. *International Conference on Acoustics.* Newton.

Hemming, C., & Lassi, M. (2010). *Copyright and the Web as Corpus.* Retrieved October 27, 2010, from Swedish National Graduate School of Language Technology: http://hemming.se/gslt/copyrightHemmingLassi.pdf

John S. Garofolo, e. a. (1993). *TIMIT Acoustic-Phonetic Continuous Speech Corpus.* Philadelphia: Linguistic Data Consortium.

Juang, B. H., & Rabiner, L. R. (2005). *Automatic Speech Recognition - A Brief History of the Technology Development.* Elsevier Encyclopedia of Language and Linguistics.

Këpuska, V. (2006). Wake-Up-Word Application for First Responder Communication Enhancement. *SPIE.* Orlando, Florida: SPIE.

Këpuska, V. (2009). *Elevator Simulator Screen Perspective.* Retrieved from YouTube: http://www.youtube.com/watch?v=OQ8eyBTbS_E;

Këpuska, V. (2009). *Elevator Simulator User Perspective*. Retrieved from YouTube: http://www.youtube.com/watch?v=j5CeVtQMvK0;

Këpuska, V. Z. (2006, January 3). *Patent No. 6983246.* USA.

Këpuska, V. Z. (2006, April 1). *Patent No. 7085717.* USA.

Këpuska, V. Z., & Klein, T. B. (2009). A novel Wake-Up-Word speech recognition system, Wake-Up-Word recognition task, technology and evaluation. *Nonlinear Analysis: Theory Methods and Applications.*, e2772–e2789.

Këpuska, V., & Shih, C. (2010). Prosodic Analysis of Alerting and Referential context of Sentinel Words. Orlando: AIPR.

Këpuska, V., Carstens, D. S., & Wallace, R. (2006). Leading and Trailing Silence in Wake-Up-Word Speech Recognition. *Industry, Engineering & Management Systems 2006.* Cocoa Beach.

Këpuska, V., Gurbuz, S., Rodriguez, W., Fiore, S., Carstens, D., Converse, P., et al. (2008). uC: Ubiquitous Collaboration Platform for Multimodal Team Interaction Support. *Journal of International Technology and Information Managment*, 264-284.

Klein, T. B. (2007). *Triple Scoring of Hidden Markov Models in Wake-Up-Word Speech Recognition.* Thessis, Florida Institute of Technology, Melbourne.

LDC, University of Pennsylvania. (2009, October 19). *Top Ten Corpora*. Retrieved September 22, 2010, from Linguistic Data Consortium: http://www.ldc.upenn.edu/Catalog/topten.jsp

MSDN. (2010). *Microsoft Speech API 5.3*. Retrieved 8 7, 2010, from MSDN: http://msdn.microsoft.com/en-us/library/ms723627%28VS.85%29.aspx

NCH Software. (2010). Switch Audio File Converter Software.

OpenCog Wiki. (2010). *RelEx Dependency Relationship Extractor*. Retrieved September 13, 2010, from OpenCog Wiki: http://wiki.opencog.org/w/RelEx

Ramdhan, R., & Beharry, X. (2009). Movie Transcript Parser.

Rojanasthien, P. (2010). *Data Collection System for Prosodic Analysis and Acoustic Model Training for Wake-Up-Word Speech Recogntion.* Thessis, Electrical and Computer Engineering, Melbourne.

Rudnicky, A. (2010). *lmtool*. Retrieved Oct 1, 2010, from Sphinx Knowledge Base Tools: http://www.speech.cs.cmu.edu/tools/lmtool.html

Sastraputera, R. (2009). *Discriminating alerting and referential context from prosodic features.* Thessis, Florida Institute of Technology, Melbourne.

Shih, C.T. (2009). *Investigation of Prosodic Features for Wake-Up-Word Speech Recognition Task.* Thessis, Florida Institute of Technology, Melbourne.

Zuggy, T. V. (2009). SubRip 1.20/1.50b: DVD Subtitles Ripper.