Comput Syst Sci & Eng (2018) 6: 429–445

International Journal of
Computer Systems
Science & Engineering

A phoneme-based approach for eliminating out-of-vocabulary problem of Turkish speech recognition using Hidden Markov Model

Erdem Yavuz[1]* **and Vedat Topuz**[2]

[1] *Istanbul Commerce University, Department of Computer Engineering, 34840, Istanbul*
[2] *Marmara University, Vocational School of Technical Sciences, 34722, Istanbul. E-mail: vtopuz@marmara.edu.tr*

Since Turkish is a morphologically productive language, it is almost impossible for a word-based recognition system to be realized to completely model Turkish language. Due to the fact that it is difficult for the system to recognize words not introduced to it in a word-based recognition system, recognition success rate drops considerably caused by out-of-vocabulary words. In this study, a speaker-dependent, phoneme-based word recognition system has been designed and implemented for Turkish Language to overcome the problem. An algorithm for finding phoneme-boundaries has been devised in order to segment the word into its phonemes. After the segmentation of words into phonemes, each phoneme is separated into different sub-groups according to its position and neighboring phonemes in that word. Generated sub-groups are represented by Hidden Markov Model, which is a statistical technique, using Mel-frequency cepstral coefficients as feature vector. Since phoneme-based approach is adopted in this study, it has been successfully achieved that many out of vocabulary words could be recognized.

Keywords: Speech recognition, Hidden Markov model, cepstral analysis, phoneme boundary

1. INTRODUCTION

Advances in digital signal processing technology prepare the ground for speech processing to be applied in different fields such as speech compression, speech quality improvement, speech synthesis, speaker verification and speech recognition [1]. Speech recognition systems can be classified according to various inherent features. For example, according to the continuity of speech, these systems can be classified into two groups such as "*isolated word recognition*" and "*continuous speech recognition*" [2]. From a different perspective, speech recognition systems can be sorted into two groups such as "*speaker-dependent recognition*" and "*speaker-independent recognition*". A speaker-independent recognition system can recognize speeches from anybody. On the other hand, speaker-dependent recognition system can only recognize the speeches of one person introduced to the system [2]. It is no doubt that speaker-independent recognition systems are more difficult to develop and realize. In addition to the classification types mentioned, a classification can be done according to length of the recognition unit. In this context, they can be classified into two general classes such as "*word-based speech recognition*" and "*subunit-based speech recognition*". Word-based speech recognition systems can be anticipated to have high success rates, as co-articulation problem, the interaction of two different sound

*Corresponding Author: E-mail: eyavuz@ticaret.edu.tr

organs in the word, is naturally ruled out. However, large vocabulary requires considerable amount of memory and computational power. This causes the training and management of the system to be demanding. Sub-unit based systems like phoneme-based speech recognition systems is, on the other hand, deal with relatively small number of speech units and consequently need low memory and computational power requirements with respect to word-based speech recognition systems. Therefore, training process of the system can be realized quickly and effectively. However, phoneme-based systems carry the overhead of pronunciation variability caused by neighboring phonemes in the word uttered. This situation adversely affects the success of phoneme-based systems [2]. Generally, success rates of phoneme-based systems are lower than word-based systems.

It is of great significance to properly determine speech recognition unit for increasing success rate of the speech recognition system. Recognition unit that brings the success is determined according to vocabulary and recognition type, whether the system is continuous speech recognition or isolated word recognition [3]. Using a recognition unit that is smaller than a word and larger than a phoneme, e.g. syllable may provide a tradeoff between the disadvantages of word-based and phoneme-based recognition systems [4]. In this study, a novel isolated word recognition system, which is categorized under the field of speech recognition is proposed for an agglutinative language.

The rest of the paper is as follows. Section 2 covers related studies presented in literature. Section 3 introduces phoneme-based word recognition model including the sub-sections relating to Hidden Markov Model (HMM) and Mel-frequency cepstral coefficient (MFCC). Results and discussion of the proposed system are given in Section 4. Finally, conclusion and future work being under study are summarized in Section 5.

2. BACKGROUND AND MOTIVATION

Few studies in literature related to speech recognition systems developed for Turkish Language until now are given below. Carki et al. proposed a study [5] related to large vocabulary continuous speech recognition system. Omitting the out of vocabulary (OOV) words, the proposed method had a word error rate (WER) of 16.9% when tested on the words introduced to the system.

A rule-based speaker-dependent recognition system for Turkish was proposed in a master thesis by Mengüşoğlu [6]. Phonemes were used as recognition units in the study. A special syllabic database was created by examining syllable structures of Turkish words. The database created was eventually used for checking purposes when converting speech to text. An effective detection algorithm for detecting syllable boundaries was suggested as a future work in the study.

A language model that can be an alternative to word-based approach in Turkish speech recognition was searched in the study [7] done by Dutağacı. In this direction, morphem-based, stem-based, syllable-based models were compared in the study. All of these three models were stated to be advantageous than word-based model in terms of vocabulary to be handled, context sensitivity and complexity. All experiments were performed on texts rather than implementing a real speech recognition system.

In a study done by Tunalı and Doğruel in the framework of a master thesis, a speaker-dependent isolated word recognition system for Turkish was implemented [8]. Phonemes were used as the smallest unit for recognition and a special phoneme boundary detection algorithm was developed. Phonemes were modelled by HMMs using features extracted by cepstral analysis. 100 words were added to the vocabulary of the developed system. 591 phonemes were modelled by HMMs as a result of 100-word vocabulary. Average response time for 5-letter words was reported to be about 25 seconds in the system. To shorten this long response time, an improvement was needed to be done in the search algorithm as reported.

In a study proposed by Can [9], a syllable-based Turkish recognition system was designed and implemented using ANN. 3500 different syllables were reported to be valid and proper for Turkish Language. Besides, it was claimed that two-letter and three-letter syllables constituted 63% of the syllables encountered in Turkish Language. In this context, a success rate of about 63% could allegedly be achieved when the syllable types mentioned were introduced.

In a work presented in the framework of a doctoral study [10], a speaker-dependent syllable-based Turkish speech recognition engine was designed and implemented using dynamic time warping (DTW), multi-layer perceptron (MLP) artificial neural network, HMM and support vector machine (SVM) methods together. Developed system consisted of 200 words indicating medium vocabulary system. In this framework, each word in the vocabulary was uttered 10 times to form a database of 2000 samples totally. WER values of system for DTW, ANN, HMM and SVM approaches were 5.8%, 12%, 17.4% and 9.2% respectively.

In the work of Tombaloğlu and Erdem [11], a Turkish speech recognition system was proposed using SVM and MFCC methods. A phoneme-based approach was adopted in the study and a database was formed accordingly. The database formed was tested on two other methods such as LPC-HMM and LPC-MFCC. Higher success rate was reported in the study for SVM method with respect to LPC-HMM method.

A speech recognition system based on tri-phone model was designed for Turkish Language presented in the study by Patlar and Akbulut [12]. Because of the agglutinative structure and suffix morphology of Turkish Language, it was preferred to form acoustic models based on tri-phones, modelled by HMMs of five-states. The aim was to refrain from the huge growth potential of vocabulary for Turkish by deriving new words by suffixing.

Recognition performances of different language models such as stem based, stem-ending and morph based and word based ones were compared for a limited audio corpus for Turkish [13, 14]. A hybrid model of word-based and stem-ending based approaches was also proposed in the study. For a limited corpus, word-based model was stated to have a better performance when compared to sub-word models.

Turkish is a language in the Ural-Altai language family. Turkish is an agglutinative language with rich suffix morphology, enabling many words to be derived in the language. Every suffix carries a different linguistic meaning/message. Due to the agglutinative structure of it, large vocabulary continuous speech recognition task is very difficult for Turkish unlike English and Japanese. Because of the morphological productivity of Turk-

ish, it is not possible for a word-based recognition system to be realized to completely model Turkish language. Since it is not possible for the system to recognize words not introduced to it in a word-based recognition system, recognition success rate drops considerably caused by these OOV words. On the contrary, success rates reaching up to 95% have been achieved for English with the same approach [15].

To solve the OOV problem, first thing that comes to mind could be to keep the number of words in the dictionary of the system as high as possible. This operation is not possible for agglutinative languages. Also, it extremely prolongs response time of the systems and it causes the system to be infeasible anymore.

Considering all the issues related to Turkish, in order to develop large vocabulary continuous speech recognition systems for Turkish, it is necessary to use different approaches other than the word-based one. In this context, phoneme-based approach is preferred to realize a speech recognition system in the presented study, as every language has a very finite number of phonemes. In this direction, a speaker-dependent phoneme-based word recognition system is designed and implemented in the study. An algorithm for finding phoneme-boundaries has been devised in order to segment the word into its phonemes. After finding phoneme boundaries, each phoneme is separated into different sub-groups according to its position and neighboring phonemes in the related word. Categorized phonemes are together modelled by 3-state HMMs. A search network consisting of HMM states and transitions is formed using MFCCs. Recognition is done on the network by a Viterbi search algorithm to find the most probable states according to the temporal transitions of the utterance. Since phoneme-based approach was adopted in this study, it has been achieved that many out of vocabulary words could be recognized.

3. METHOD

3.1 Mel Frequency Cepstral Coefficients

Psychophysical studies show that human ear do not perceive frequency contents in a linear scale [16]. Studies have proven that this scale is linear up to 1 kHz, and logarithmic above 1 kHz. The function that corresponds to this scale order and converts frequency axis to the proper form to be processed by Mel cepstral analysis [16, 17] is expressed as

$$\Omega(\omega) = 2595 \log_{10}\left[1 + \frac{\omega}{1400\pi}\right] \quad (1)$$

where ω denotes the frequency as rad/sec.

Inspired by the psychophysical studies related to the frequency discrimination capability of human ear, filters and their critical bandwidths are spaced in an incremental manner. Figure 1 shows triangular filters [18] proposed by Davis and Mermelstein used for converting Fourier outputs to Mel scale [1]. Generally, 20 filters are used for speech signals of 4 kHz bandwidth and a few additional filters are used for signals with higher bandwidth.

The differences between the spectral peaks are reduced by nonlinear transformation of the critical band power spectrum [16]. In cepstral analysis, compression operation mentioned is done by taking the logarithm of power spectrum. In cepstral analysis, discrete cosine transform converts auditory spectrum-like structure to real coefficients. These coefficients are factors of cosine terms which results from decomposition of the compressed spectrum. In the Mel cepstrum, these coefficients correspond to amplitudes of Fourier components of the logarithmic spectrum [16]. Generally, a spectrum smoothing operation is done in the computation of MFCCs. In the phase of critical band integration, some smoothing operation is inherently done on the spectrum suppressing its considerable details. Additionally, a separate smoothing operation is necessary to lessen the effects of factors not related to the linguistic message at this stage. Generally, this operation is done by truncating first 12 or 14 components out of, for example, 20 filter outputs [16].

Mathematical equation of MFCCs is expressed as

$$c_j = \sqrt{\frac{2}{N}} \sum_{i=1}^{N} A_i \cos\left(\frac{\pi j (i - 0.5)}{N}\right) \; 0 \le j < N \quad (2)$$

where c_j denotes jth cepstral coefficient and A_i denotes logarithmic amplitudes for N channels. N represents the number of filters in the filter-bank. In general, 13 static parameters are totally obtained by including energy term to the first 12 coefficients. For the purpose of improving robustness and stability of speech recognition systems, dynamic features are incorporated in order to include the information related to temporal dependency between successive analysis frames [16]. For this reason, first-order time derivatives (known as "deltas"), and second-order time derivatives (known as "delta-deltas") may be added to the coefficients. Time derivative coefficients are tend to be noisy due to their sensitivity to random fluctuations on original static features. In order to more robustly measure variations, linear regression is generally performed on analysis frames [1]. After all, delta and delta-delta coefficients may be appended to the static parameters (MFCCs) to form a 39-parameter feature vector.

3.2 Hidden Markov Model

Hidden Markov Model (HMM) is a very effective statistical method to model the changes in the dynamics of successive feature vectors. HMM consists of states, transitions of states and some related parameters. HMMs can be used to represent statistical attributes of speech sub-units such as word, syllable and phoneme [16]. The assumption underlying the statistical modelling is that a signal can be defined as a parametric random process and the parameters of this random process can be well defined in a deterministic manner [19]. The reason that HMM is a very effective and popular method is rested on its powerful mathematical foundations. Detailed information about mathematical foundations, and evaluation, estimation and training processes of HMM can be found in the studies [16, 19-20], respectively. Structure of a 3-state HMM example is illustrated in Figure 2.

It is assumed that the HMM in the figure represents a word consisting of three stationary parts. Acoustic analysis of utterances is done for that word and then computed feature vectors are used in order to train the model for that word. Models formed in the training process are then used as a reference point in recognition process. Instead of comparing with words in the database, it is more advantageous to adopt a statistical model, as statistics

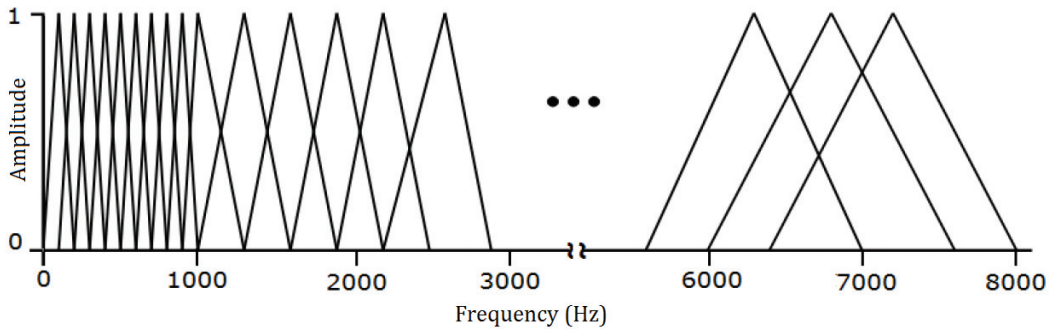


Figure 1 Triangular filters for converting Fourier outputs into Mel scale.

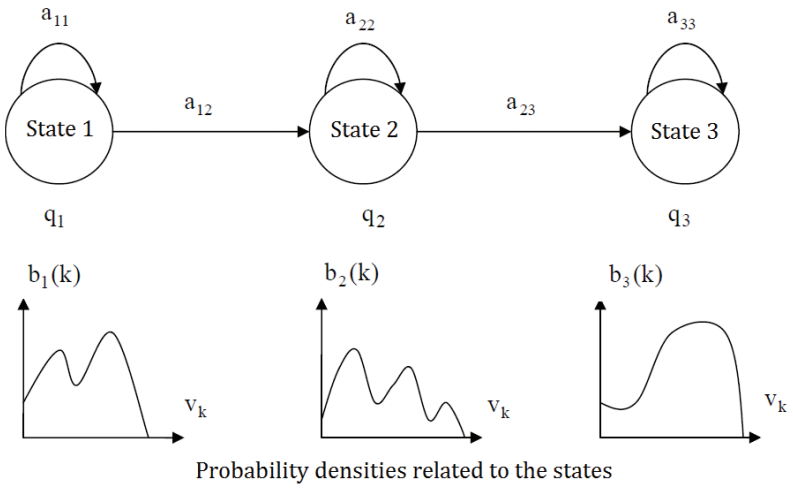


Figure 2 Example of a HMM with 3-states.

generated from many previous samples is better at generalization of new samples.

For example, the Turkish word "merhaba" (meaning 'hello' in English) is composed of the phonemes /m/ - /e/ - /r/ - /h/ - /a/ - /b/ - /a/. This structure can be easily modelled by a left-right HMM and every state in the model corresponds to a phone in the speech. It can be inferred that each phoneme in a word carry characteristic feature values in a certain probability density. Similarly, a phoneme itself can be modelled by left-right HMM in some applications. For instance, the transient part in which a phoneme is affected by the preceding phoneme, the stationary part of that phoneme, and the transient part in which the succeeding phoneme begins to have influence on. By the way, this structure of HMM is also adopted in our study.

3.3 Proposed Recognition System

Speech recognition system developed in the study consists of two main stages, 'training' and 'recognition' processes. In addition to MATLAB's internal toolboxes, two external toolboxes presented under General Public Licence are used in this study. First one is a library named "H2M" [21] which contains many functions related to HMM. The second is the library named "VOICEBOX" [22] which contains speech processing functions like MELCEPST function used for calculating MFCCs.

Since the speech recognition system in the study is speaker-dependent, it is necessary for the user to utter several times the words that the system is expected to recognize. Speech recordings are processed and features are extracted in order to be used in successive stages. Then, phoneme boundaries are detected by a novel algorithm devised for the phoneme-based recognition approach. Phonemes are separated into sub-groups according to their positions and neighboring phonemes in the word. Then, phonemes in each of these sub-groups are used for creating a HMM for phonemes (indeed the variant of a phoneme). Training process steps verbalized in this paragraph is depicted in the flowchart shown in Figure 3 and elaborated under the next four sub-headings.

3.3.1 Speech Recording

It is required that the user utter a word several times in the developed recognition system. Accumulating more data for a phoneme provides that the system makes better generalization for that phoneme. The words introduced to the system in the implementation and test process is given in the table in Appendix A. Word utterances are recorded at a sampling rate of 11025 Hz as 16-bit mono wave audio using "Microsoft° LifeChat® LX-2000" headset.

3.3.2 Feature Extraction

Due to its physiological nature, shape of human vocal tract changes quite slowly [1]. From this point of view, speech sig-

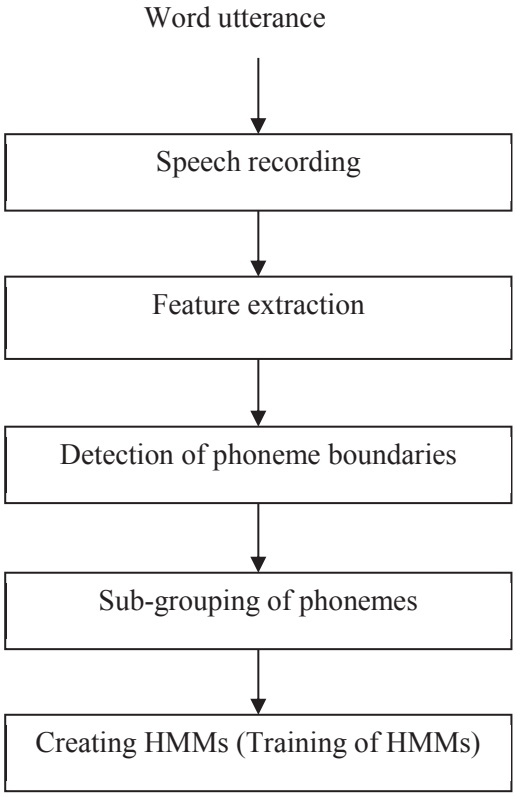


Figure 3 The flowchart demonstrating the steps of training process.

nal is generally split into analysis frames of 20-32ms length. In order to increase frame numbers for a limited duration speech signal, the span between starting times of successive frames are kept smaller than the frame length, i.e. successive frames contain some shared data. This is called overlapping.

Speech signal is split into frames of 20 ms duration. Since the sampling frequency is 11025 Hz, the frame in the study contains 220 samples (discrete values). Moreover, overlapping ratio is 50%, so analysis frames are created by progressing in 10 ms steps. Length of analysis frame and overlapping ratio relates to HMM creation parameters mentioned in the next chapters.

Speech signal in analysis frames should be represented by less data, i.e. features should be extracted from the signal. Feature extraction method adopted in the system is based on cepstral analysis producing MFCCs. Details of computing MFCCs are explained previously in Section 3.1. Including first-order and second-order time derivatives, a feature vector of 75 coefficients is constructed by processing each frame. Extracted features are used later in the stages of HMM creation and recognition.

3.3.3 Algorithm for Determining Phoneme Boundaries

In order to achieve phoneme-based modeling of uttered words, it is first required to determine phoneme boundaries. Unfortunately, there is no silver bullet for separating phonemes. This is indeed quite complex problem and stands as a research field [23]. An introductory algorithm is proposed to find phoneme boundaries in this study. It is able to find phonemes of the uttered words of the form that is a vowel followed by a consonant or vice versa. Flowchart of the algorithm is given in Figure 4.

First of all, signal of the uttered word is split into frames of 20 ms durations with an overlapping ratio of 50%. Obtained frames are multiplied by a hamming window in order to get rid of discontinuities at each end of the frame which cause unwanted fictitious spikes in the frequency domain.

Later, spectral envelope of each frame is computed. Burg auto-regressive (AR) method is used for this operation. The method (computed by 50^{th} order) is run on Fast Fourier Transform (FFT) bins of 256 (window length).

Line fitting operation is done using spectral envelope data computed for each frame. Line fitting is based upon the least squares method. N points (data) is fitted on a line model according to the formula given in Eq. 3.

$$y(x) = y(x; a, b) = ax + b \tag{3}$$

Line fitting is done by minimizing the chi-square function expressed as

$$X^2(a, b) = \sum_{i=1}^{N} \left(\frac{y_i a x_i b}{\sigma_i} \right) \tag{4}$$

By minimizing Eq. 4, the values a(corresponding to the slope term) and b(corresponding to the constant term) are obtained. In this study, the constant term, i.e. the value b is used.

Spectral envelope data of a frame give some clue about characteristics of the signal portion in that frame. When data reflecting the characteristics are thought to be scattered in the Cartesian coordinate systems, the thing we want to do is to find the line best fitting the data. Slope of the fitted line or its points crossing the axes give information about which class that the frame examined belongs to, i.e. vowel or consonant phoneme. Figure 5 shows an example of line fitting results for vowel and consonant phoneme frames respectively. Solid and dotted lines in the

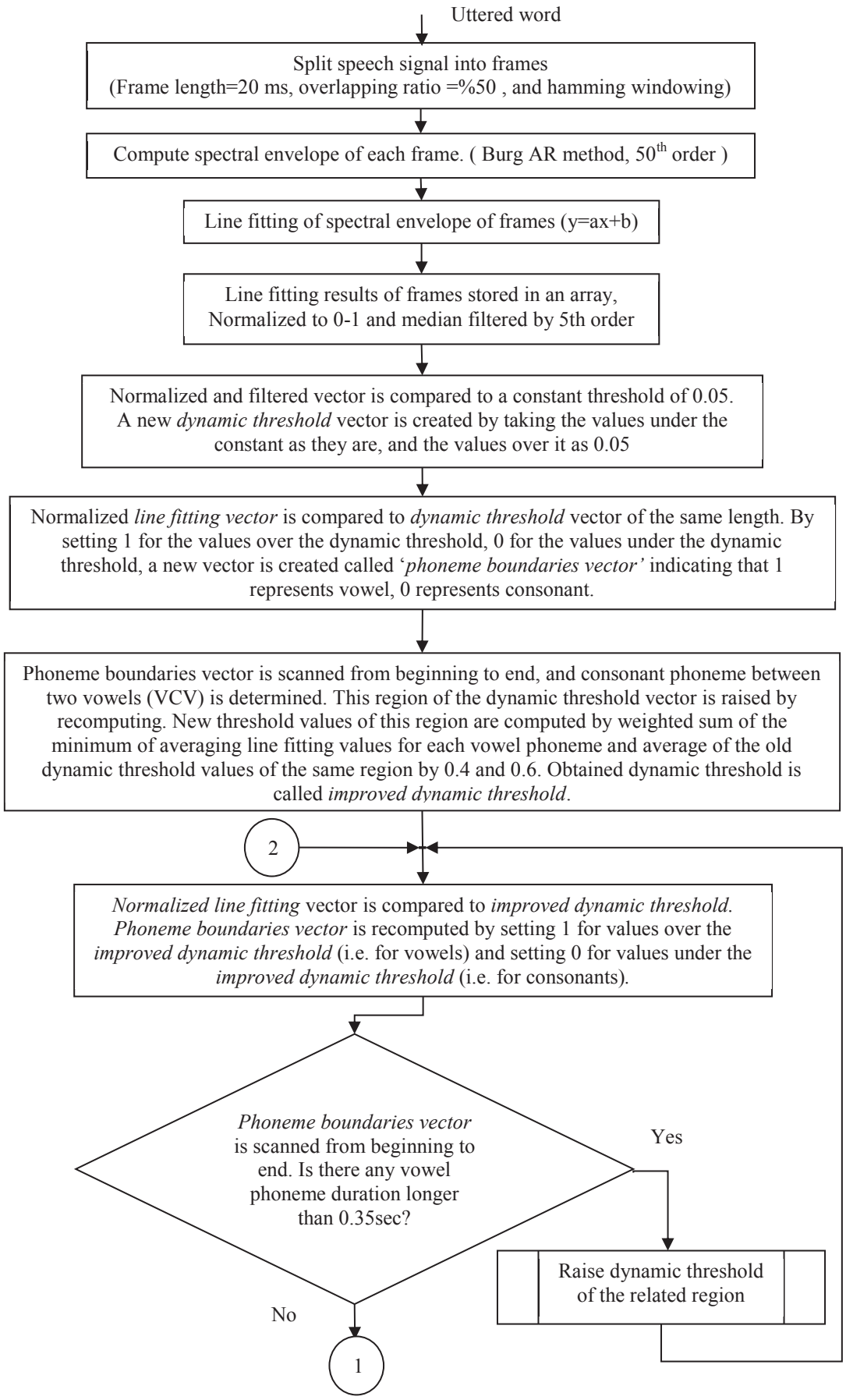


Figure 4 Flowchart of the algorithm for determining phoneme boundaries.

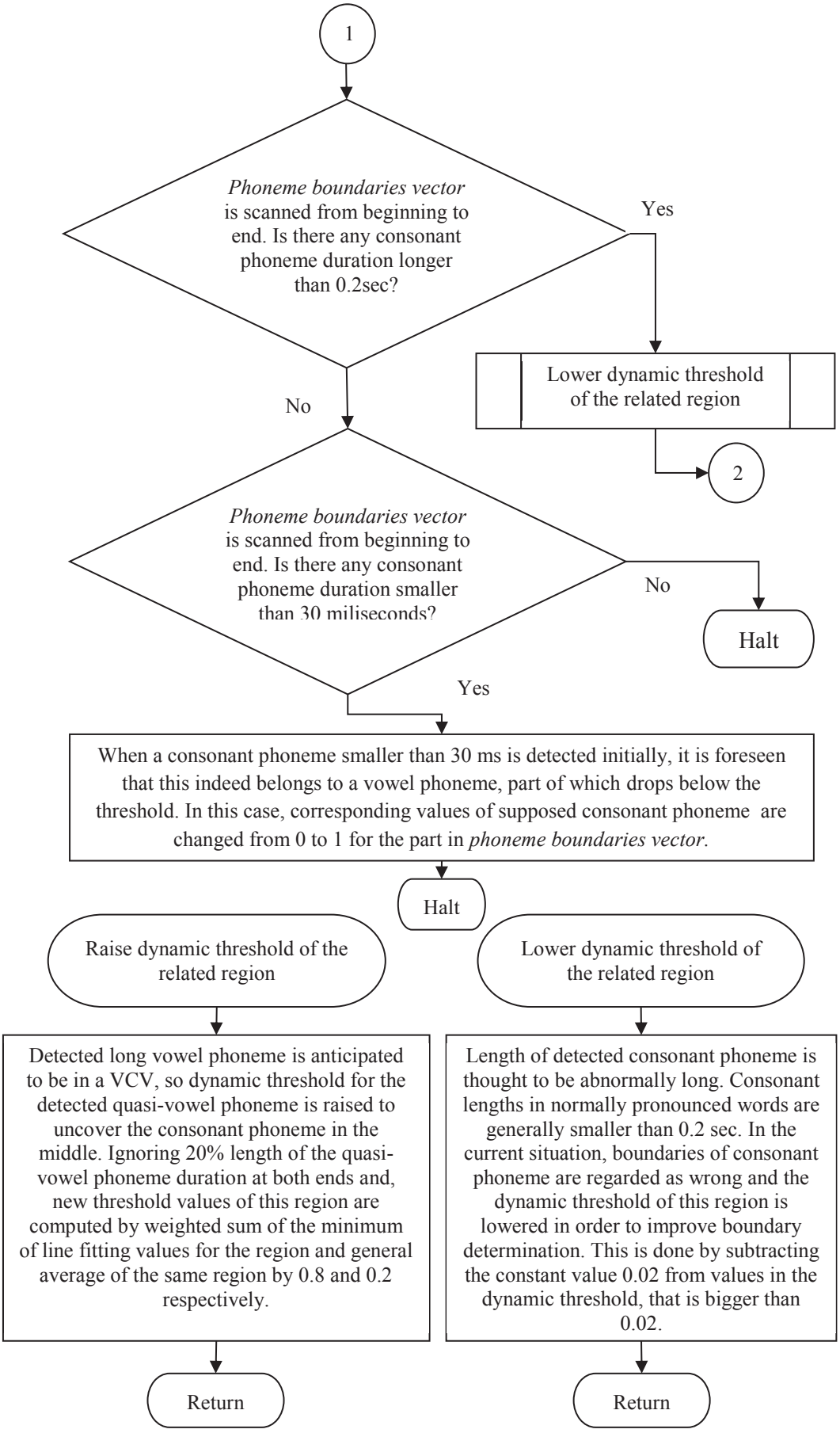


Figure 4 (continued) Flowchart of the algorithm for determining phoneme boundaries.

figure represents spectral envelope and fitted line respectively. Values in the axes do not indicate real values. Instead, fitted line is multiplied by a scaler in order to properly visualize it. It is easily noticed from the figure that magnitudes of the slope and constant terms for vowel phoneme frames are generally much larger than those of consonant phoneme frames. This is used for phoneme discrimination between vowel and consonant frames.

Line fitting is performed for all frames to form a vector. Obtained vector is normalized to 0-1 and then smoothed by a 5^{th} order median filter in order to suppress sudden changes. Resultantly, normalized and filtered vector contains values close to 1 for vowel phonemes and values close to 0 for consonant phonemes. As the rest of the algorithm is descriptively given in the flowchart in Figures 3, it is opted for not verbally expressing the rest of it again in here. Instead, intermediate results and final output of the phoneme boundary determination algorithm for an example word is given in Figure 6 and stated in the next paragraph.

Uttered example word that we want to determine phoneme boundaries is the Turkish word "*acaba*", waveform of which is plotted in Figure 6a. Looking at the waveform, one can easily infer that amplitude of vowel phonemes are generally larger than that of consonants. However, comparing averages of amplitude values is not enough to discriminate between vowel and consonant phonemes. Some consonant phonemes may have as high amplitudes as vowel phonemes.

For example, at first glance one can expect the period between 0.12 and 0.23 seconds to correspond to the second phoneme "*a*" in the word "*acaba*". As a matter of fact, if it was done by exploiting the average of signal amplitudes, it would most probably be deduced that the period corresponded to the second "*a*" phoneme in that word. When examined by any audio speech analysis tool, it is figured out that second "a" phoneme in that word indeed corresponds to the period between 0.16 and 0.23 seconds. Normalized version of the line fitting result and its median filter result are plotted in Figure 6b. *Dynamic threshold* used in finding phoneme boundaries is shown in Figure 6c as dotted line. As seen from the figure, threshold of the region between two vowels should be raised to accurately determine boundaries of the consonant in the middle. This operation is done by the 7^{th} block of the flowchart in Figure 4 and the result, called *improved dynamic threshold* is shown as dotted line in Figure 6d. Line fitting result computed before is compared to this new threshold to give *phoneme boundaries vector* plotted as red solid line in Figure 6e. The constants used for weighting factors in the flowchart have been empirically determined as optimal values as a result of repetitive trials. As they are treated on normalized vectors, it is seen that phoneme boundaries vector generates stable and consistent results for different speakers. Vowel and consonant phonemes correspond to 1 and 0 respectively in this vector. Eventually, phoneme boundaries vector is used to detect and separate phonemes in the word utterance, resulting in five phonemes plotted separately in Figure 7.

3.3.4 Formation of Phoneme Sub-groups

The aim at this stage is to separate phonemes in the uttered words into different sub-groups according to its position and neighboring phonemes in the related word. There exist variants of a phoneme depending on its position and neighboring phonemes in that word. Sub-groups are created for each phoneme according to this position and neighborhood, and phonemes with the same characteristics standing in different words are accumulated in the same sub-group. Each created sub-group is then used to create a HMM explained in the next section. The goal of creating sub-groups of phonemes is to avoid deterioration of generalization capability caused by variabilities of a phoneme having different neighboring phonemes and different position.

Task of creating phoneme sub-groups is illustrated in Figure 8 and done in the following way:

- Phonemes of the uttered word in the training process are handled in order and each phoneme is saved in a separate file.
- Each vowel phoneme handled is called with the name of preceding consonant phoneme or silence sign ("-") if it is first phoneme in the uttered word. All phonemes are accumulated in the related folders named accordingly. For example, the vowel phonemes of the word "*acaba*" shown in Figure 7 are separated and put into folders in this way : "-a", "ca", "ba". These separated vowel phonemes are accumulated in the folders named "Phoneme _a_-**a**", "Phoneme _a_c**a**", "Phoneme _a_b**a**" respectively.
- Each consonant phoneme handled is called with the names of preceding and succeeding vowel phonemes and accumulated in that name folder. For example, the consonant phonemes of the word "*acaba*" shown in Figure 7 are separated and put into folders in this way : "-a**c**a^1", "a**b**a^2". These separated consonant phonemes are accumulated in the folders named "Phoneme_c_a**c**a^1", "Phoneme _b_a**b**a^2", respectively.

Figure 8 shows three words "acaba", "adada" and "edada" and resultant 10 sub-groups for 5 different phonemes after sub-grouping process. If phonemes were classified directly not by considering neighboring phonemes, 5 groups would be created totally. Since a system to be built in this structure would include variants of the same phoneme in the same class, the HMM model created in this way becomes more difficult to generalize properly and is likely to produce erroneous results. To illustrate this situation, it is seen that first consonant phoneme of the word "*adada*" is affected by the phoneme "a". Likewise, first consonant phoneme of the word "*edada*" is affected by the phoneme "e". Including these two derivatives of the same phoneme in the same class would cause the phone to be characterized by a single HMM, thereby reducing the success rate in the recognition phase.

3.3.5 Training of HMMs

After creating sub-groups of phonemes, each sub-group is modelled by a separate HMM. Structure of HMM adopted in the study is given in Figure 9.

Modelling phonemes is realized by left-to-right HMM structure, in which there exists transitions between HMM states from left to right depending on the observation sequence. This structure is very suitable for modelling speech signals. When speech signal data is examined, it is very normal that the signal characteristic changes as the time progresses. The same is also true

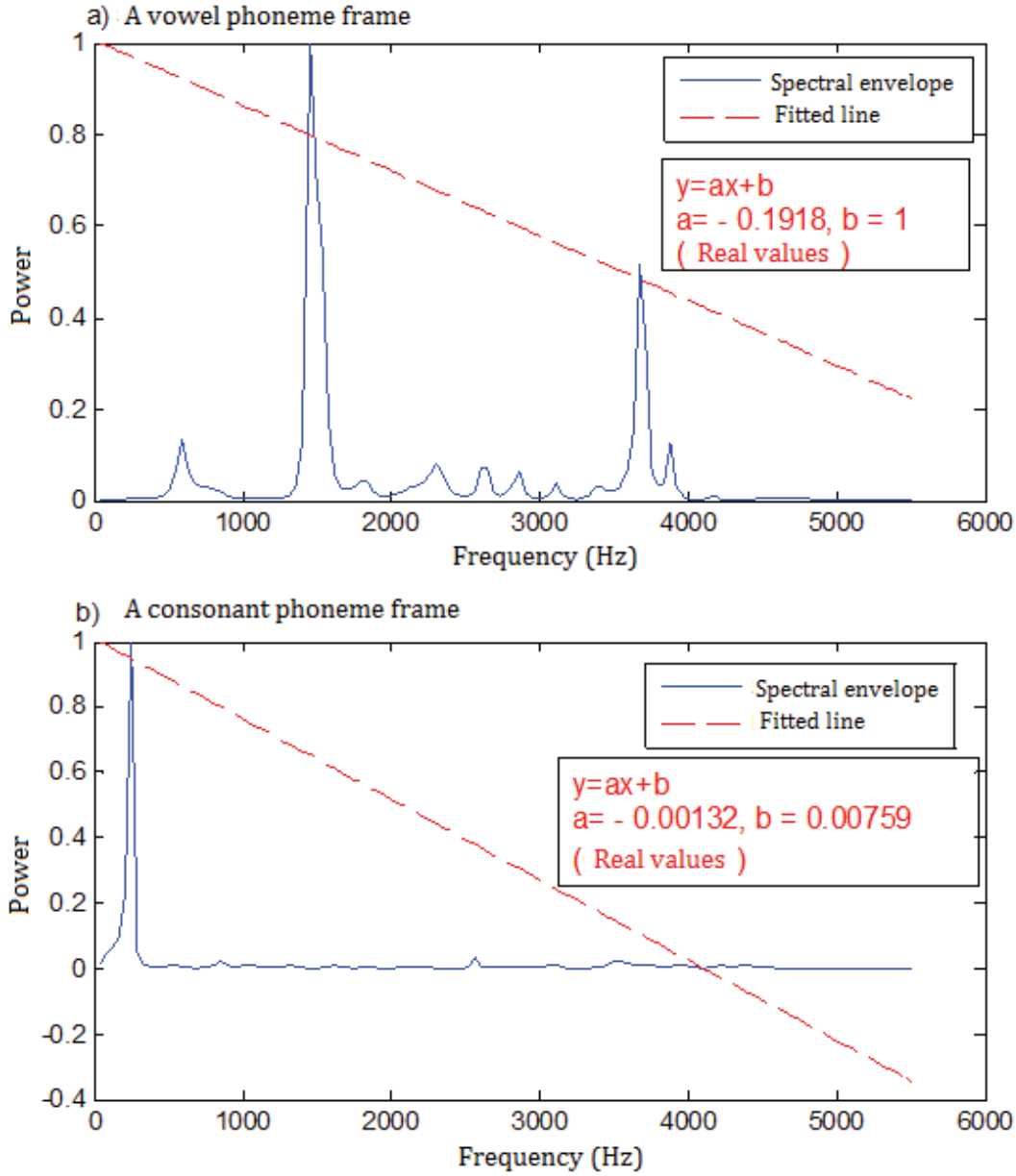


Figure 5 Example frame and its fitted line (a) for vowel phoneme and (b) for consonant phoneme.

for single phoneme signal, because signal characteristics in the starting phase, stationary phase, and ending phase of a phoneme are naturally different from each other. For this reason, 3-state HMM structure is adopted in the presented study.

Two parameters called *mu* (matrix that holds the average of distributions depending on the states) and *sigma* (covariance matrix) are kept for each HMM created by "hmm_mint" function [20] in the aforementioned H2M library. Size of *mu* parameter is 3x75, as 3 rows for 3 states and 75 coefficients from cepstral analysis. Similarly, size of *sigma* parameter is 1x75.

3.3.6 Recognition process

Recognition in this study is a process in which a text transcription is generated corresponding to the word uttered using the previously trained phoneme models. Stages of recognition process are depicted in Figure 10. First three stages of the recognition process are the same as those of the training process shown in Figure 3. Fourth and last stage is phoneme matching stage explained in the next paragraph.

In the phoneme matching phase, phonemes of the word to be recognized is identified one by making a Viterbi search among the phoneme models generated during the training process. Then, identified phonemes are put together to bring out the word uttered. An HMM that best represents the phone desired to be identified gets the highest score out of all the trained phonemes and it gives its own name to the phone (or phoneme) desired to be identified. Repeated letters (phonemes) may appear in the recognized words because of the error margin in the phoneme boundary detection algorithm. As mentioned earlier, in this study, it is aimed to identify a vowel phoneme followed by a consonant phoneme or vice versa. Therefore, if there are repetitive letters

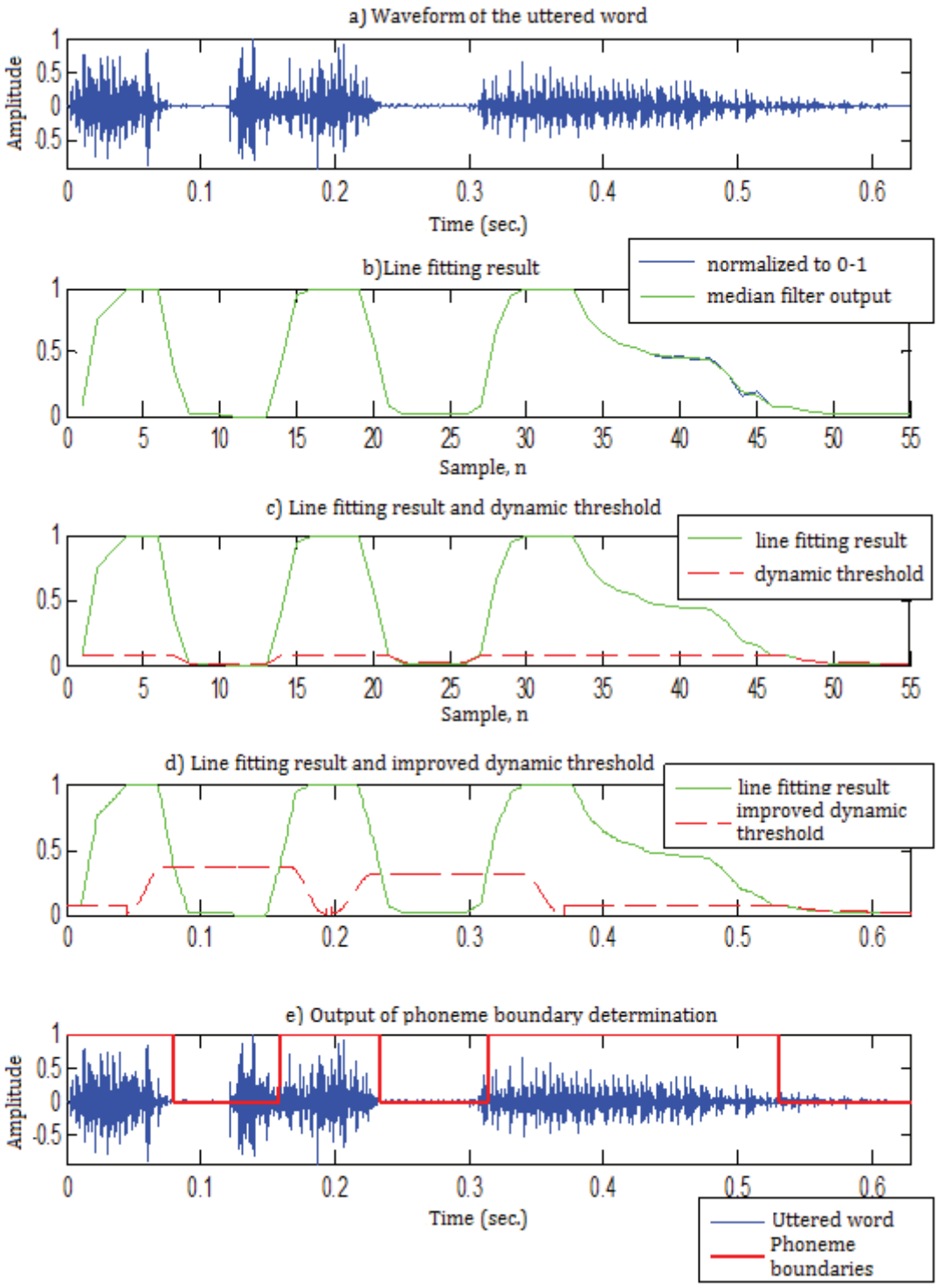


Figure 6 Intermediate results and final output of the phoneme boundary determination algorithm for the Turkish word "acaba".

in the initial results produced by the system, redundant ones are omitted and thrown away. Table 1 shows outputs of the system for some word utterances.

4. RESULTS AND DISCUSSION

To be introduced to phoneme-based word recognition system, 40 different words, each of which is uttered 10 times have been recorded. These words are given in the table in Appendix A. After forming sub-groups of phonemes, 140 different HMMs have been created for all these words lasting for averagely 40 seconds.

Several tests have been carried out by changing various parameters of the system in order to attain the best recognition performance. These tests were performed in three types, namely the length of the feature vector and the feature vector components, the number of states in the HMM model used, and the length of

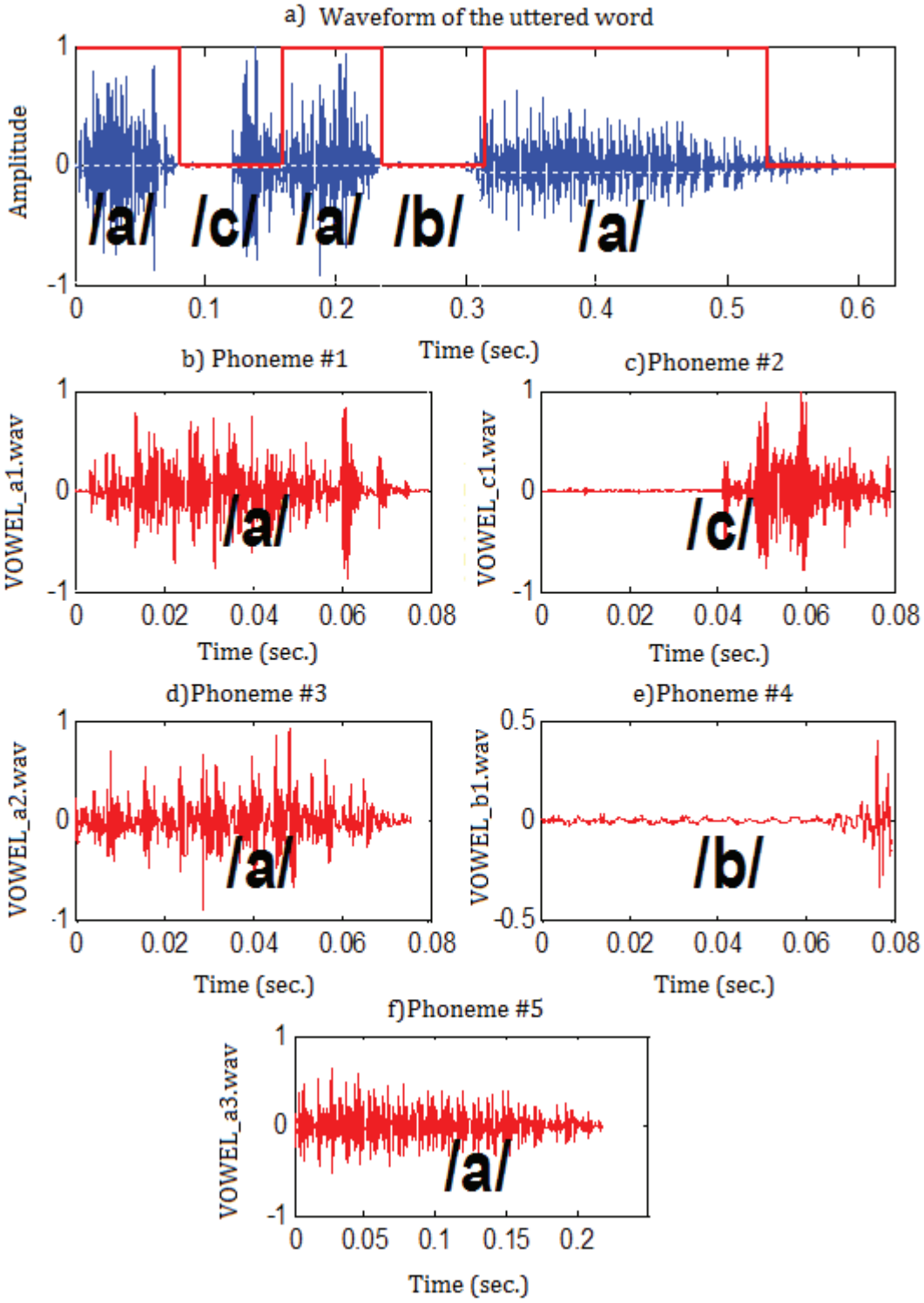
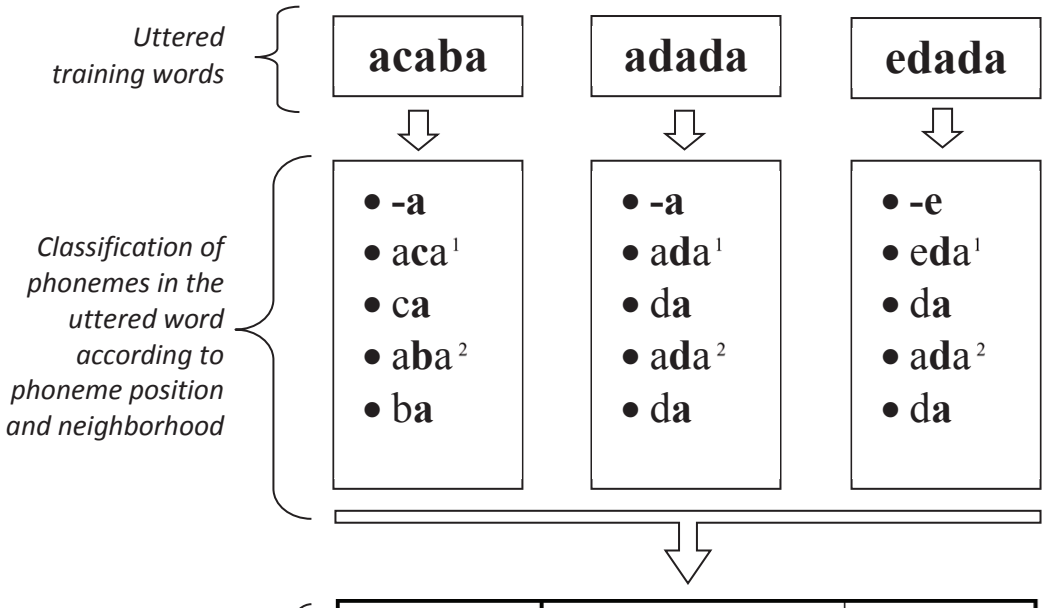


Figure 7 Separate phonemes detected after determining phoneme boundaries of the uttered word "*acaba*".

the analysis frame. Tests were done on 10 different words uttered at another time. It should be noted that these ten words are the words contained in the system's vocabulary. In order to see the effect of the parameters more clearly, these ten words were used consistently in the three different types of tests mentioned.

The results obtained from the tests done by changing the length of the feature vector are given in Table 2. It is known that the time derivative coefficients (i.e. dynamic parameters) must be added to the MFCC feature vector in order to improve the stability and success of the recognition system. The result of the first test confirms this fact. The removal of dynamic parameters from the feature vector leads to misidentification of some phonemes as seen in Table 2, causing a slight decrease in the recognition success. On the other hand, changing the length of the feature vector does not make a significant difference on the success of the system, provided that the static and dynamic components



Sub-groups of phonemes resulted from classification according to phoneme position and neighborhood and sample number in each sub-group

Phonemes	*Phoneme Sub-group*	*(Training) Sample number*
/a/	Phoneme_a_-a	2
	Phoneme _a_ba	1
	Phoneme _a_ca	1
	Phoneme _a_da	4
/b/	Phoneme _b_aba2	1
/c/	Phoneme _c_aca1	1
/d/	Phoneme _d_ada1	1
	Phoneme _d_ada2	2
	Phoneme _d_eda1	1
/e/	Phoneme _e_-e	1

Figure 8 Diagram illustrating the creation of sub-groups.

are preserved. In the experiments using the coefficient vectors of 39 (13 +13 +13) and 75 (25 + 25 + 25) coefficients, all of the ten word utterances were identified in the same way by the system. In addition, as the size of the feature vector increases, the duration of HMM model creation also increases. On the other hand, length of the feature vector has almost no effect on recognition time.

The second test, results of which are tabulated in Table 3 shows the effect of the number of states in the HMM model on the system. As seen from the table, number of HMM states has no effect on recognition success. However, increasing the number of cases has led to a significant increase in total HMM building time. For example, when 5-state HMM models are used instead of 2-state HMM models, the model building time has increased by about 30%.

The third test, results of which is given in Table 4 is related to the effects of frame length and overlapping ratio on the recognition success. Test results have shown that frame length of 15-25 ms is more suitable for phoneme-based speech recognition systems. When the frame length is too short (e.g. 10 ms frame length and 50% overlapping ratio in Table 4), identification of some phonemes may be incorrect because the analysis

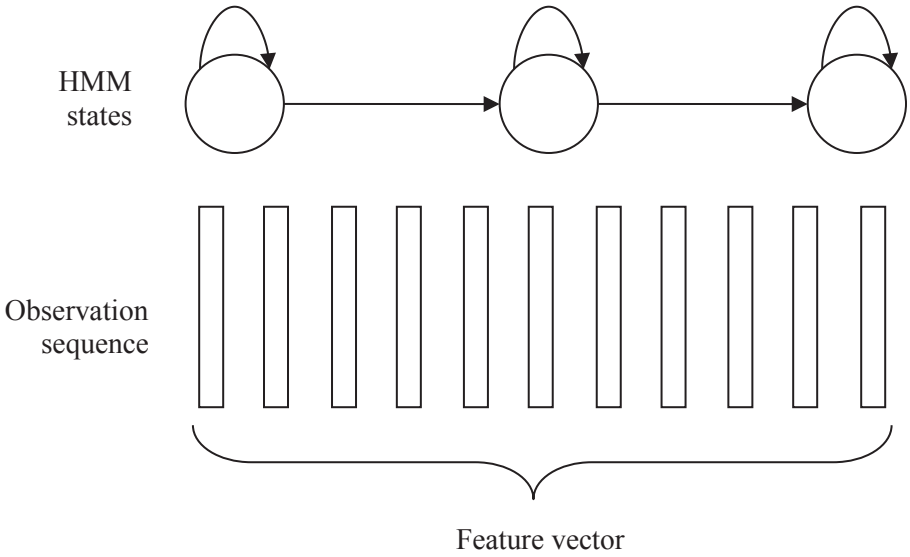


Figure 9 HMM structure used for modelling phonemes.

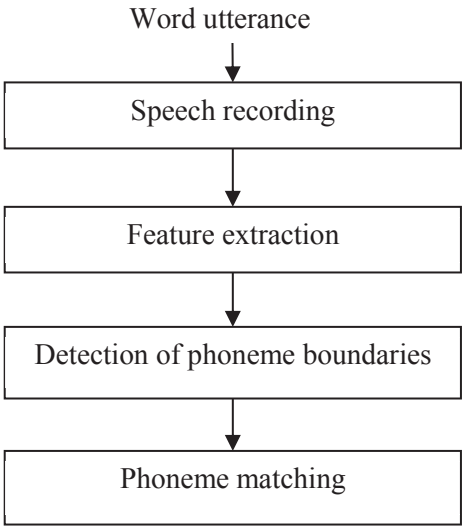


Figure 10 Stages of recognition process.

Table 1 Outputs generated by the system for some word utterances.

Uttered word	Initial results of the system	Final result after improvement
acaba	acabaa	acaba
adana	adanaa	adana
ceket	ceket	ceket
çikolata	çikolataa	çikolata
esasen	esasen	esasen
ezine	ezine	ezine
limonata	liimonata	limonata
ötekisine	ötekisine	ötekisine
salata	salata	salata
sevilen	sevileen	sevilen

Table 2 Results of the test done by changing the length and components of the feature vector.

HMM state number=3					
Frame length=20 ms, overlapping ratio=80%					
	Feature vector:	*39 coefficients (MFCCs+ delta + delta-delta)*	*75 coefficients (MFCCs+ delta + delta-delta)*	*50 coefficients (MFCCs+ delta)*	*25 coefficients (only MFCCs)*
	Uttered word	*Recognition result*	*Recognition result*	*Recognition result*	*Recognition result*
1	*acaba*	**acaba**	**acaba**	**acaba**	**acaba**
2	*akide*	**akide**	**akide**	**akide**	**akide**
3	*azize*	**azize**	**azize**	**azize**	**azize**
4	*ceket*	**ceket**	**ceket**	**ceket**	**ceket**
5	*çikolata*	**çikolata**	**çikolata**	**çikolata**	**çikolata**
6	*esasen*	**esasen**	**esasen**	**esasen**	**esasen**
7	*evimize*	**evimize**	**evimize**	**evimize**	**evimize**
8	*limonata*	**limonata**	**limonata**	**limonata**	**limonata**
9	*sevilen*	**sevilen**	**sevilen**	semilen	semilen
10	*sepet*	**sepet**	**sepet**	**sepet**	**sepet**
HMM model creating time (*for all the training words in the vocabulary*) (second)		**38,85**	**43,189**	**40,03**	**37,46**
Total recognition time (*for 10 words*) (second)		**20,02**	**20,02**	**19,09**	**19,06**

Table 3 Results of the test done by changing the number of HMM states.

Feature vector: 75 Coefficient (MFCCs+ delta + delta-delta)					
Frame length=15 ms, overlapping ratio=80%					
	Feature vector:	$N = 2$	$N = 3$	$N = 4$	$N = 5$
	Uttered word	*Recognition result*	*Recognition result*	*Recognition result*	*Recognition result*
1	*acaba*	**acaba**	**acaba**	**acaba**	**acaba**
2	*akide*	**akide**	**akide**	**akide**	**akide**
3	*azize*	**azize**	**azize**	**azize**	**azize**
4	*ceket*	**ceket**	**ceket**	**ceket**	**ceket**
5	*çikolata*	**çikolata**	**çikolata**	**çikolata**	**çikolata**
6	*esasen*	**esasen**	**esasen**	**esasen**	**esasen**
7	*evimize*	**evimize**	**evimize**	**evimize**	**evimize**
8	*limonata*	**limonata**	**limonata**	**limonata**	**limonata**
9	*sevilen*	**sevilen**	**sevilen**	**sevilen**	**sevilen**
10	*sepet*	**sepet**	**sepet**	**sepet**	**sepet**
HMM model creating time (*for all the training words in the vocabulary*) (second)		**54,18**	**59,28**	**63,9**	**70,05**
Total recognition time (*for 10 words*) (second)		**20,4**	**20,7**	**21,07**	**21,05**

frame does not contain enough speech data. Conversely when the length of the analysis frame is too long (e.g. 35 ms frame length in Table 4 and 85% overlapping), some phonemes that are rapidly changing within themselves may be misinterpreted as it will be difficult to catch these changes. Meanwhile, the change in the length of the analysis frame has not had a significant effect on the response time of the system.

From the findings obtained from these tests, optimum running

Table 4 Results of the test done by changing the frame length and overlapping ratio.

HMM state number=3					
Feature vector: 75 Coefficient (MFCCs+ delta + delta-delta)					
	Feature vector:	*20 ms, 80%*	*15 ms, 80%*	*10 ms, 50%*	*35 ms, 85%*
	Uttered word	*Recognition result*	*Recognition result*	*Recognition result*	*Recognition result*
1	*acaba*	**acaba**	**acaba**	**acaba**	**acaba**
2	*akide*	**akide**	**akide**	akile	**akide**
3	*azize*	**azize**	**azize**	**azize**	**azize**
4	*ceket*	**ceket**	**ceket**	**ceket**	**ceket**
5	*çikolata*	**çikolata**	**çikolata**	**çikolata**	**çikolata**
6	*esasen*	**esasen**	**esasen**	**esasen**	esase
7	*evimize*	**evimize**	**evimize**	**evimize**	**evimize**
8	*limonata*	**limonata**	**limonata**	**limonata**	**limonata**
9	*sevilen*	**sevilen**	**sevilen**	**sevilen**	sevilea
10	*sepet*	**sepet**	**sepet**	**sepet**	**sepet**
HMM model creating time (*for all the training words in the vocabulary*) (second)		**43,189**	**59,28**	**34,7**	**47,16**
Total recognition time (*for 10 words*) (second)		**20,02**	**20,7**	**19,59**	**20,68**

parameters have been determined for developed word recognition system. A feature vector of 75-coefficient has been used to include both static and dynamic components. The frame length is 20 ms and the overlapping ratio is 80%, and the phonemes are modelled with a 3-state HMM structure. After setting the specified parameters, each of the words in the dictionary of the system were recorded with careful pronunciation, again 10 times, in order to test the performance of the system. Recognition results of only a few test words are included in Appendix B (because of page limit). It has been seen from the test results that the system is very successful in recognizing these words. The system takes an average of 2 seconds to recognize a word with 5-6 letters.

It has been observed that realized phoneme-based recognition system is able to identify many words that are not introduced to the system (i.e., not found in the vocabulary of the system, OOV words). The 48 words listed in Appendix C are some of the words that can be identified by the system even though they are not introduced to it. The speech recognition system was implemented on a PC with Pentium Core2Duro 2.4 GHz processor and Windows Vista operating system.

5. CONCLUSION

In this paper, a novel design and implementation for speaker-dependent phoneme-based word recognition system is proposed. A new algorithm is devised to determine phoneme boundaries to detect phonemes of uttered words. Phonemes are modelled by 3-state HMMs using MFCC feature vectors. By courtesy of its phoneme-based structure, it is possible to identify and recognize many words that are not in the system's dictionary (OOV words) as well as the words introduced to the system. It takes for the system to identify a 5-6 letter word for 2 seconds averagely. The most important factor that makes the response time of the system relatively short is the adopted phoneme-based approach. While there are numerous words (individual speech units) that should be introduced to the system in a word-based approach, the number of individual units is very limited in the adopted approach. At this point, the model space to be searched using the Viterbi algorithm has been endeavored to be kept as small as possible.

Developed algorithm for now is able to find phonemes of the uttered words of the form that is a vowel followed by a consonant or vice versa. By eliminating this deficiency, a more robust and dynamic structure of the algorithm will increase the success rate by allowing different words to be added to the vocabulary of the system. For example, using more intelligent methods such as support vector machines, the algorithm for detecting phoneme boundaries can be made more robust. As a future work, various grammatical rules and syllable statistics obtained from written sources can be introduced to improve the recognition success.

REFERENCES

1. Holmes, W (2001). *Speech synthesis and recognition*. London, CRC press.
2. Furui, S (2000). Digital Speech Processing, Synthesis, and Recognition. *Digital Speech Processing, Synthesis, and Recognition (Second Edition, Revised and Expanded)*.
3. Rabiner, L and Juang, BH (2008). Historical Perspective of the Field of ASR/NLU. In *Springer Handbook of Speech Processing* (pp. 521-538). Springer Berlin Heidelberg.

4. Lippmann, RP (1989). Review of neural networks for speech recognition. *Neural computation*, 1(1), 1-38.
5. Carki, K, Geutner, P, and Schultz, T (2000). Turkish LVCSR: towards better speech recognition for agglutinative languages. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on* (Vol. 3, pp. 1563-1566). IEEE.
6. Mengüsoglu, E (1999). Rule Based Design and Implementation of a Speech Recognition System for Turkish Language. *Master of Science Thesis, Hacettepe University, Institute of Applied Sciences*. Ankara, Turkey.
7. Dutağacı, H (2002). Statistical language models for large vocabulary Turkish speech recognition. *Doctoral dissertation, Boğaziçi University*. Istanbul, Turkey.
8. Tunali, V and Doğruel, M (2005). A speaker dependent, large vocabulary, isolated word speech recognition system for Turkish. *Master of Science Thesis, Marmara University*. Istanbul, Turkey.
9. Can, B (2007). Design and implementation of a syllable-based Turkish speech recognition system. *Master of Science Thesis, Hacettepe University, Institute of Applied Sciences*. Ankara, Turkey.
10. Aşlıyan, R (2008). Design and Implementation of Turkish Recognition Engine, *Ph.D. Thesis*, *Dokuz Eylül University*, *Graduate Schools of Natural and Applied Sciences*, İzmir, Turkey.
11. Tombaloğlu, B, Erdem, H (2016). Development of a MFCC-SVM Based Turkish Speech Recognition system. In *Signal Processing and Communication Application Conference (SIU), 2016 24th* (pp. 929-932). IEEE
12. Patlar, F, and Akbulut, A (2010). Triphone Continuous Speech Recognition System for Turkish Language Using Hidden Markov Model. In *Proceedings of the 12th IASTED International Conference* (Vol. 710, No. 059, p. 13).
13. Susman, D, Köprü, S, and Yazici, A (2012). Turkish Large Vocabulary Continuous Speech Recognition by using limited audio corpus. In *2012 20th Signal Processing and Communications Applications Conference (SIU)* (pp. 1-4). IEEE.
14. Köprü, S (2012). Turkish Large Vocabulary Continuous Speech Recognition By Using Limited Audio Corpus .*Doctoral dissertation, Middle East Technical University*. Ankara, Turkey.
15. Büyük, O (2005). Sub-word language modeling for Turkish speech recognition. *Doctoral dissertation, Sabanci University*. Istanbul, Turkey.
16. Mogran, N, Bourlard, H, and Hermansky, H (2004). Automatic speech recognition: An auditory perspective. In *Speech processing in the auditory system* (pp. 309-338). Springer New York.
17. O'shaughnessy, D (1987). Speech communication: human and machine. Universities press. Addison-Wesley Publ. Comp.
18. Yavuz, E, Topuz, V (2010). Recognition of Turkish vowels by probabilistic neural networks using yule-walker AR method. In *International Conference on Hybrid Artificial Intelligence Systems* (pp. 112-119). Springer Berlin Heidelberg.
19. Rabiner, LR (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 257-286.
20. Zheng, X, Li, J, Zhang, Y, and Liu, Q (2016). An optimization model of Hadoop cluster performance prediction based on Markov process. *Computer System Science and Engineering*, 31(2), 127-136.
21. Cappé, O (2001). H2M: A set of MATLAB/OCTAVE functions for the EM estimation of mixtures and hidden Markov models. *URL: https://perso.limsi.fr/cappe/Code/H2m/Documentation/*
22. sBrookes, M. (2003). VOICEBOX: a MATLAB toolbox for speech processing. URL: *www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html*
23. Niu, Y, Xiong, M, Guo, J, Mao, C, Xian, Y, et al. (2016). Using cross ambiguity model improves the effect of vietnamese word segmentation. *Computer System Science and Engineering*, 31(6), 475-484.

Appendix

A. List of the words introduced to the system

abide	cemile	evimize	önümüze
acaba	çikolata	ezine	ötekisi
adana	ecele	fesane	öteleme
akide	egede	lekeli	özel
aleme	ekonomi	limon	salata
asabi	elaleme	limonata	sepet
atalet	eleman	lokomotif	sevilen
atama	emanet	odun	vasat
azize	esasen	okula	yeter
ceket	etiket	otomatik	zafer

B. Recognition results of a few test words (in vocabulary) (for Feature vector=75 coefficients, frame length=20 ms, overlapping ratio=%80 and HMM state number=3)

Tested word : **abide**	*Test Word Sample #*	*Recognition result*	Tested word: **acaba**	*Test Word Sample #*	*Recognition result*	Tested word: **adana**	*Test Word Sample #*	*Recognition result*
	1	abide		1	acaba		1	adana
	2	abide		2	acaba		2	adana
	3	abide		3	acaba		3	adana
	4	abide		4	acaba		4	adana
	5	abide		5	acaba		5	adana
	6	abidet		6	acaba		6	adana
	7	abide		7	acaba		7	adana
	8	abide		8	acaba		8	adana
	9	abide		9	acaba		9	adana
	10	abile		10	acaba		10	adana
Tested word: **akide**	*Test Word Sample #*	*Recognition result*	Tested word: **aleme**	*Test Word Sample #*	*Recognition result*	Tested word: **asabi**	*Test Word Sample #*	*Recognition result*
	1	akide		1	aleme		1	asabi
	2	akidet		2	aleme		2	asabi
	3	akide		3	aleme		3	asabi
	4	akiden		4	aleme		4	asabik
	5	akide		5	aieme		5	asabi
	6	akide		6	alemei		6	asabi
	7	akide		7	aleme		7	asabit
	8	akide		8	aleme		8	asabi
	9	akile		9	aleme		9	asabi
	10	akide		10	aleme		10	asabi
Tested word: **atalet**	*Test Word Sample #*	*Recognition result*	Tested word: **atama**	*Test Word Sample #*	*Recognition result*	Tested word: **akide**	*Test Word Sample #*	*Recognition result*
	1	atalet		1	atama		1	azize
	2	atalet		2	atama		2	azize
	3	atalet		3	atama		3	azize
	4	atalet		4	atama		4	azize
	5	atalet		5	atama		5	azize
	6	atalet		6	ataman		6	azize
	7	atalet		7	atama		7	azize
	8	atalet		8	atama		8	azize
	9	atalet		9	atama		9	azize
	10	atalet		10	atama		10	azize

C. List of the words not introduced to the system

abide	cemile	evimize	önümüze
acaba	çikolata	ezine	ötekisi
adana	ecele	fesane	öteleme
akide	egede	lekeli	özel
aleme	ekonomi	limon	salata
asabi	elaleme	limonata	sepet
atalet	eleman	lokomotif	sevilen
atama	emanet	odun	vasat
azize	esasen	okula	yeter
ceket	etiket	otomatik	zafer