13

Syllable Based Speech Recognition

Rıfat Aşlıyan
Adnan Menderes University
Turkey

1. Introduction

Speech recognition is the process of converting speech signals to the text. Studies on speech recognition have increased very fast for the last twenty-five years. Most of these studies have used phoneme and word as speech recognition units. Namely, in phoneme based speech recognition systems, all phonemes in a language have been modelled by a speech recognition method, and then the phonemes can be detected by these models. The recognized words are constructed with concatenating these phonemes. However, word based systems model the word utterances and try to recognize the word as a text unit. Word based systems have better success rate than phoneme based systems. As a rule, if a speech recognition system has longer recognition unit than sub-word units, it has better success rate on recognition process. In addition, phoneme end-point detection is quite difficult operation, and this effects the success of the system.

Turkish, that is one of the least studied languages in the speech recognition field, has different characteristics than European languages which require different language modelling technique. Since Turkish is an agglutinative language, the degree of inflection is very high. So, many words are generated from a Turkish word's root by adding suffixes. That's why, word based speech recognition systems are not adequate for large scaled Turkish speech recognition. Because Turkish is a syllabified language and there are approximately 3500 different Turkish syllables, speech recognition system for Turkish will be suitable to use syllable (sub-word unit) as a speech recognition unit.

Turkish speech recognition studies have increased in the past decade. These studies were based on self organizing feature map (Artuner, 1994), DTW (Meral, 1996; Özkan, 1997), HMM (Arısoy & Dutağacı, 2006; Karaca, 1999; Koç, 2002; Salor & Pellom, 2007; Yılmaz, 1999) and Discrete Wavelet Neural Network (DWNN) and Multi-layer Perceptron (MLP) (Avcı, 2007).

In a simplified way, speech recognizer includes the operations as preprocessing, feature extraction, training, recognition and postprocessing. After the speech recognizer takes the acoustic speech signal as an input, the output of the recognizer will be the recognized text.

The most common approaches to speech recognition can be divided into two classes: "template based approach" and "model based approach". Template based approaches as Dynamic Time Warping (DTW) are the simplest techniques and have the highest accuracy when used properly. The electrical signal from the microphone is digitized by an analog-to-digital converter. The system attempts to match the input with a digitized voice sample, or template. The system contains the input template, and attempts to match this template with

the actual input. Model based approaches as Artificial Neural Network (ANN), Hidden Markov Model (HMM) and Support Vector Machine (SVM) tend to extract robust representations of the speech references in a statistical way from huge amounts of speech data. Model based approaches are currently the most popular techniques. However, when the size of the vocabulary is small and the amount of training data is limited, template based approaches are still very attractive.

Speech recognition system can be speaker dependent or speaker independent. A speaker dependent system is developed to operate for a single speaker. These systems are usually more successful. A speaker independent system is developed to operate for any speaker. These systems are the most difficult to develop, most expensive and accuracy is lower than speaker dependent systems.

The size of vocabulary of a speech recognition system affects the complexity, processing requirements and the accuracy of the system. Some applications only require a few words such as only numbers; others require very large dictionaries such as dictation machines. According to vocabulary size, speech recognition systems can be divided into three main categories as small vocabulary recognizers (smaller than 100 words), medium vocabulary recognizers (around 100-1000 words) and large vocabulary recognizers (over 1000 words).

Speech recognition studies started in the late 1940s. Dreyfus-Graf (1952) designed his first "Phonetographe". This system transcribed speech into phonetic "atoms". Davis et al. (1952) designed the first speaker dependent, isolated digit recognizer. This system used a limited number of acoustic parameters based on zero-crossing counting. In Bell Laboratories, a word recognizer is designed with a phonetic decoding approach using phonetic units (Dudley & Balashek, 1958). Jakobson et al. (1952) developed the speaker independent recognition of vowels. At RCA laboratories, phonetic approach was used in the first "phonetic typewriter" which recognizes syllables by a single speaker (Olson & Belar, 1956).

The first experiments on computer based speech recognition were started in the late 1950s. The studies as speaker independent recognition of ten vowels (Forgie & Forgie, 1959), phoneme identification (Sakai & Doshita, 1962), and digit recognition (Nagata et al., 1963) have been made at this period.

Dynamic time warping using dynamic programming was proposed by Russian researchers (Slutsker, 1968; Vintsyuk, 1968). Reddy (1966) developed continuous speech recognition by dynamic tracking of phonemes.

Several systems such as HARPY (Lowerre, 1976), HEARSAY II (Lesser et al., 1975), HWIM (Wolf & Woods, 1977), MYRTILLE I (Haton & Pierrel, 1976) and KEAL (Mercier, 1977) have been implemented in 1970s.

Statistical modelling methods as Hidden Markov Models (HMM) (Ferguson, 1980; Rabiner, 1989) were used in 1980s. Neural networks as perceptron were proposed in 1950s. In 1980s, neural network approaches were presented again (Lippmann, 1987).

Support Vector Machine (SVM) was presented as a novel method for solving pattern recognition problems (Blanz et al., 1996; Cortes & Vapnik, 1995; Osuna et al., 1997). SVM has been successfully used for implementation speech recognition systems in 1990s and 2000s.

This chapter is organized as follows. In the next section, system databases, preprocessing operation, word and syllable end-point detection, feature extraction, dynamic time warping, artificial neural network and postprocessing operation have been explained. In Section 3, how to detect misspelled words using syllable *n*-gram frequencies is presented in detail. The experimental results of the developed systems have been given in Section 4. The final section concludes the chapter.

2. Implementation of syllable based speech recognition

In this study, speaker dependent isolated word speech recognition systems using DTW and ANN have been designed and implemented, and the speech signal features as mfcc, lpc, parcor, cepstrum, rasta and the mixture of mfcc, lpc and parcor have been used for the speech recognition approaches.
The speech recognition applications have been executed on the computer which has the following features: Pentium Centrino 1.6 CPU, 768 MB RAM, 40 GB harddisk, Windows XP Operating System, a sound card and a microphone. The codes of the applications have been written with Matlab 6.5.

2.1 System databases

System dictionary consists of 200 different Turkish words (Aşlıyan, 2010). Using this dictionary, two databases have been constructed (Aşlıyan, 2010). One database has been used for training and the other is for testing of the system.
The training speech database (approximately 2.7 hours of 250 MB speech material) involves 5000 Turkish word utterances (25x200) which were recorded by a male speaker. Each word in the dictionary was recorded 25 times.
The testing speech database was constructed by recording every word 10 times in the dictionary. Total number of utterances is 2000 (about 1.1 hours of 100 MB speech material).
The recording procedure took place in a noise-free environment. A head-mounted close-talking microphone was used. The format of the file recording is WAVE file format. The waveforms of the utterances are encoded using Pulse Code Modulation (PCM) coding format, 11025 Hz sampling rate, 2 bytes per sample. The utterances are recorded in 2 seconds time duration.

2.2 Preprocessing operation

After the digitization of the word speech signal, preemphasis filter has been applied to spectrally flatten the signal. For the speech signal the syllable end-point detection is applied. After that each syllable utterance is divided into frames of 20 ms by frameblocking. To reduce the signal discontinuity at the ends of each block, Hamming window is applied for each frame.

2.3 Word and syllable end-point detection

An important problem in speech processing is to detect the presence of speech in a background of noise. This problem is often referred to as the end-point location problem (Rabiner & Sambur, 1975). The accurate detection of a word's start and end-points means that subsequent processing of the data can be kept to a minimum.
A major cause of errors in isolated-word automatic speech recognition systems is the inaccurate detection of the beginning and ending boundaries of test and reference patterns (Junqua et al., 1997). It is essential for automatic speech recognition algorithms that speech segments are reliably separated from non-speech.
The reason for requiring an effective end-point algorithm is that the computation for processing the speech is minimum when the end-points are accurately located (Savoji, 1989).
In syllable end-point detection operation, the speech signals are taken and after processing them, the number of syllables and the indexes of syllable end-points have been detected. Namely, the beginning and end indexes are computed on the digital speech signal.

After sampling the sound wav files, the mean of the speech signal as a vector is calculated and translated to $y=0$ axis. Assume that y_n is a speech signal. The new speech signal, which is focused on $y=0$ axis, is $y'_n = y_n - mean(y_n)$. After that, the voiced and unvoiced parts of the speech signal are approximately computed with the slope between the beginning value of the digital sound and the maximum value of the sound. This slope is the threshold slope. The utterance is divided into windows which have 350 samples. If the slope, which is calculated between two windows, which are one after the other, is greater than the threshold slope, this means that the voiced part of the sound begins at the index. However, these beginning and end index of the voiced part are nearly true, but not certain value. The distance data of zero-crossing index of sound vector has been used because of obtaining more accurate results. A new vector which represents the zero-crossing distances, has been constructed, then a threshold has been defined (say zero-crossing threshold=100). The beginning index is found earlier but not certainly true index. Now this index goes on one by one to the first index if the zero-crossing distance is between 1 and zero-crossing threshold. Zero values of the vector are not taken into account. In the same way, the end index of the voiced part is calculated. At the end, the voiced part is found exactly.

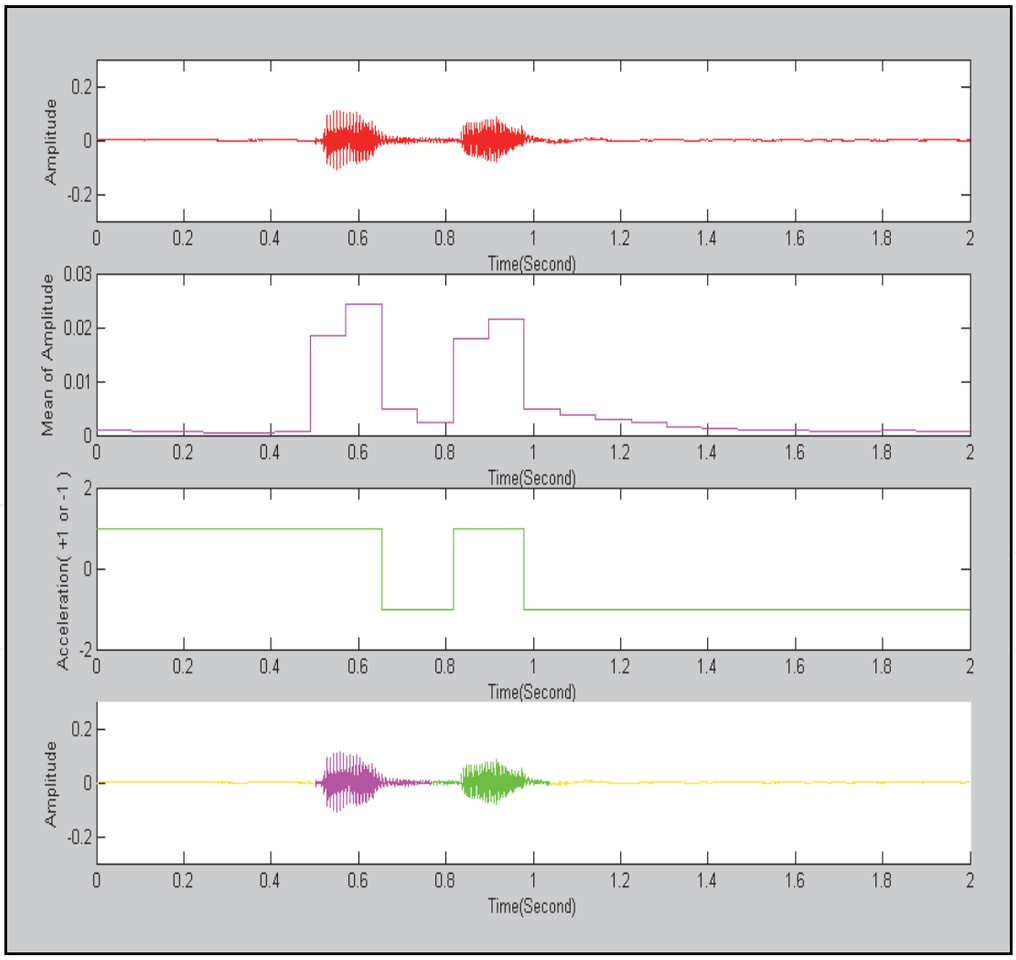

Fig. 1. The process of syllable endpoint detection

To discover syllable end-points, the windows that consist of 900 samples are generated without overlapping. The mean values of these windows are computed and assembled for constructing a new mean vector as shown in Figure 1. The slope between one element and the next element of the mean vector is determined, and if the slope is zero or greater than zero, a new vector's value is +1. Otherwise, the value is -1. Using these vectors, the boundaries of syllables on the sound vector are obtained approximately. The samples between 500 samples backward and 500 samples forward from the found syllable end-points are divided into windows which include 20 samples. After that, the middle index of the window which has the minimum mean is syllable end-point. Finally, the beginning and end index of the syllables can be calculated for each word before processing them. Now we have the number of syllables of the word and their end-point indexes.
According to the number of syllables in a word using syllable end-point detection algorithm which is mentioned in Subsection 2.3.2, it is found that the accuracy result is approximately 99%. For example, the word which has five syllables is successfully divided into five syllables and the end-points of the syllables are detected.

2.3.1 Word end-point detection algorithm

Step 1. x in Equation 1 is digital sound vector. $N = 22050$ (N is the number of samples in the utterance)

$$x = (x_1, x_2, x_3 ..., x_N) \tag{1}$$

Step 2. λ is the mean of the values of first 200 samples in x. $\tilde{x}$ is a vector which translated to the axis $y = 0$.

$$\lambda = (\sum_{i=1}^{200} x_i) / 200 \tag{2}$$

$$\tilde{x} = x - \lambda = (x_1 - \lambda, ..., x_N - \lambda) = (\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_N) \tag{3}$$

Step 3. M is the maximum value of the vector $\tilde{x}$. I is the index of maximum value of the vector $\tilde{x}$. E_b and E_s are the beginning and end threshold values respectively.

$$[M, I] = \max(\tilde{x}) \tag{4}$$

$$E_b = M / I \quad E_s = M / (N - I) \tag{5}$$

Step 4. The vector $\tilde{x}$ is divided into windows which consist of 350 samples. The vector $\overline{x}$ is the mean vector of above windows.

$$\overline{x} = (\overline{x}_1, \overline{x}_2, ..., \overline{x}_p) \text{ and } p = N / 350 \tag{6}$$

$$\overline{x}_i = \left(\sum_{k=i*350}^{(i+1)*350-1} \tilde{x}_k \right) / 350 \ , i = 1, 2, ..., p \tag{7}$$

Step 5. For $i = 1,2,...,p-1$, $\overline{x}_E$ and $\overline{x}_{E_i}$ are calculated as shown in Equation 8.

$$\overline{x}_E = (\overline{x}_{E_1}, \overline{x}_{E_2}, ..., \overline{x}_{E_{p-1}}) \text{ and } \overline{x}_{E_i} = \overline{x}_{i+1} / \overline{x}_i \tag{8}$$

Step 6. S_b is the beginning index of the sound vector.

For $r = 1$ to $p-1$

if $\overline{x}_{E_r} > E_b$ then $S_b = r * 350$

End

Step 7. S_s is the end index of the sound vector.

For r = $x_{E_{p-1}}$ DownTo 1

if $1 / \overline{x}_{E_r} > E_s$ then $S_s = r * 350$

End

Step 8. The beginning and end indexes are approximately determined from Step 6 and 7. To decide exactly the end-points of the sound, the zero-crossing indexes are fixed. Using the sound vector $\tilde{x} = (\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_N)$, the zero-crossing vector $z = (z_1, z_2, ..., z_{N-1})$ is generated.

For $k = 2$ To N

if $\tilde{x}_{k-1} / \tilde{x}_k < 0$ then

$z_{k-1} = 1$

else

$z_{k-1} = 0$

End

Step 9. After the distances between one after the other zero-crossing indexes are computed, new distance vector of zero-crossing $\tilde{z}_k = (\tilde{z}_1, \tilde{z}_2, ..., \tilde{z}_{N-1})$ is calculated as the followings.

For $k = 1$ To $N-1$

if $z_k = 1$ and after the index k, its first value of the following indexes is 1,

($z_h = 1$) then

$\tilde{z}_k = h - k$

else

$\tilde{z}_k = 0$

if $z_k = 0$ then $\tilde{z}_k = 0$

End

Step 10. The threshold value of zero-crossing is accepted as $T = 100$. SB is the value at the index which the sound begins.

$SB = S_b$

For $k = S_b$ DownTo 1

if $\tilde{z}_k > 0$ and $\tilde{z}_k < T$ then $SB = k$

if $\tilde{z}_k = 0$ then continue

if $\tilde{z}_k > T$ then break

End

Step 11. SS is the value at the index which the sound ends.
$SS = S_s$
For $k = S_s$ To $N-1$
if $\tilde{z}_k > 0$ and $\tilde{z}_k < T$ then $SS = k$
if $\tilde{z}_k = 0$ then continue
if $\tilde{z}_k > T$ then break
End

2.3.2 Syllable end-point detection of the word utterances

After detecting the end-points (SB and SS) of the words, the end-points of the syllables in the words are determined with the following algorithm.

Step 1. $n = (n_1, n_2, ..., n_k) = (\tilde{x}_{SB}, \tilde{x}_{SB+1}, ..., \tilde{x}_{SS})$

Step 2. The vector n is divided into windows, which have 900 samples, without overlapping. The vector $\overline{n}$ is the mean vector of each window above.
$\overline{n} = (\overline{n}_1, \overline{n}_2, ..., \overline{n}_p)$ and $p = k / 900$

$$\overline{n}_i = \left(\sum_{m=i*900}^{(i+1)*900-1} n_m \right) / 900 \, , i = 1, 2, ...p \tag{9}$$

Step 3. The slope vector is composed by computing the slopes between the values of the vector $\overline{n}$ which follow one after another.

$$\overline{n}_E = (\overline{n}_{E_1}, \overline{n}_{E_2}, ..., \overline{n}_{E_{p-1}}) \text{ and } \overline{n}_{E_i} = \overline{n}_{i+1} / \overline{n}_i \, , i = 1, 2, ..., p-1 \tag{10}$$

Step 4. Using the slope vector, we calculate the vector $a = (a_1, a_2, ..., a_{p-1})$ which has the values, +1 and -1. Namely, the increasing and decreasing positions are determined.
For $k = 1$ To $p-1$
if $n_{E_k} \geq 0$ then $a_k = 1$
else $a_k = -1$
End

Step 5. H is the number of syllables in the word.
$H = 0$
For $k = 2$ To $p-1$
if $a_{k-1} = 1$ and $a_k = -1$ then $H = H + 1$
End

Step 6. The values of the middle indexes, which include the value -1 in the vector a, are approximately the end-points of syllables. There are $H-1$ syllable end-points. The syllable end-point vector $s = (s_1, s_2, ..., s_{H-1})$ is calculated as shown in the following. The values s_i are the indexes which are the values of the vector $\tilde{x}$.

For $k = 1$ To $H-1$
if the middle index of the indexes, which have the *k*-th value -1 that follows one after the other, is w then

$s_k = SB + 900 * w$

End

Step 7. Until now, the beginning point SB and end point SS is detected. The vector s represents the end-points of syllables. To find the syllable end-points more accurately the following algorithm is performed, and the vector $\tilde{s} = (\tilde{s}_1, \tilde{s}_2, ..., \tilde{s}_{H+1})$ is attained.

$\tilde{s}_1 = SB$ and $\tilde{s}_{H+1} = SS$

For $i = 1$ To $H - 1$

The windows with 20 samples between $s_i - 500$ and $s_i + 500$ are constructed, and the mean values are computed for each windows.

if the middle index of the window, which has the smallest mean, is q

then

$\tilde{s}_{i+1} = q$

End

Step 8. The vector $\tilde{s}$ which represents the syllables end-points in the word sound vector $\tilde{x}$ is decided accurately. There are H syllables in the word. The beginning index of k-th syllable is $\tilde{s}_k$ and the end index is $\tilde{s}_{k+1}$.

2.4 Feature extraction

After preprocessing the speech signal, we have the syllable end-points of the word. The syllable utterances are framed with Hamming window. The length of one frame is 20 ms with 10 ms shift (overlapping time=10 ms). 10 features are computed from each frame. These features are lpc, parcor, cepstrum, mfcc and rasta. The number of frames is not constant, but the number of frames is normalized to 30 frames with the length of 10 as shown in Figure 2. The normalized features are used only for the speech recognition method as ANN. $s(n)$ is the syllable feature vector. $x(10, m)$ is the syllable feature matrix. For normalized features, m is 30 as illustrated in Figure 2. In Figure 3, the time duration for each speech feature is shown, and the fastest speech feature extraction algorithm among these features is mfcc.

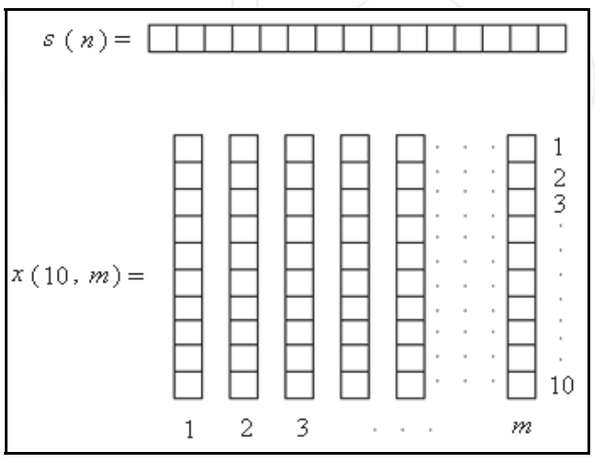

Fig. 2. Feature extraction

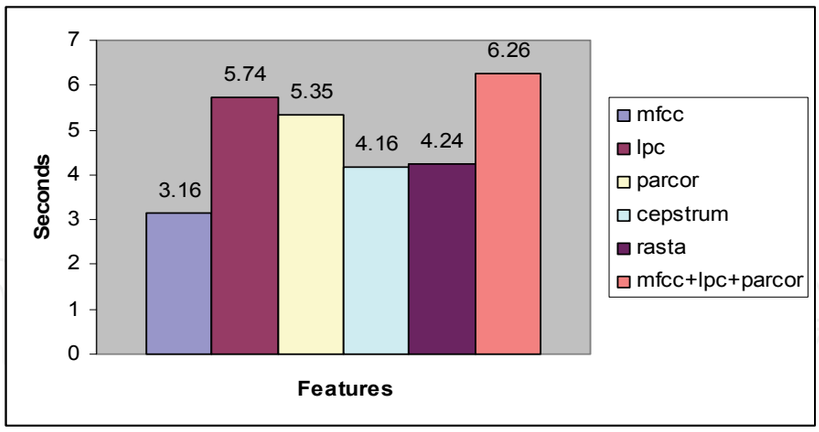

Fig. 3. Feature extraction time duration

2.5 Dynamic Time Warping

Dynamic Time Warping (DTW) (Fornés et al., 2010; Myers et al., 1980 ; Sankoff & Kruskal, 1999) is a template-based classification technique which has the principle of matching a input speech signal converted into a feature matrix against reference templates. The templates are simply feature matrix examples of each syllable of a word or syllable in the vocabulary of the system. Consequently, DTW is normally used for recognition of isolated words or syllables. The similarity between a template and the unknown speech feature matrix is assumed to be inversely proportional to the minimum cost alignment. This is normally evaluated by calculating a local distance between each input features and all feature matrices of the reference template. Calculating the minimum cost alignment is then a matter of finding the optimal path from the bottom left-hand to the top right-hand corner of the matrix. Namely, the path that accumulates the lowest sum of local distances and does not stray too far away from the diagonal.

If we define two time series $S = s_1, s_2, ..., s_n$ and $T = t_1, t_2, ..., t_m$, then DTWCost(S, T) calculates the distance between these two time series (S and T). Using dynamic programming DTW[i, j] is computed as DTW[i, j] = C + Minimum(DTW[i-1, j], DTW[i, j-1], DTW[i-1, j-1]) where C= Distance(s_i, t_j). As shown in the following, the warping path is found by detecting the minimum cost index pairs (i, j) form (0, 0) to (n, m).

2.5.1 Dynamic Time Warping algorithm

The following algorithm takes two speech feature matrices (inputs) to calculate the distance of them. The output of this algorithm is the distance of these two features. These two matrices consist of n and m frames respectively. Each frame has been assumed to get ten speech features. ∞ stands for infinity.

DTW_Cost (S[1..10, 1..n], T[1..10, 1..m])
  DTW[0..n, 0..m] as a matrix variable
  i, j, *Cost* as integer variables
  For i=1 To m
    DTW[0, i] = ∞
  end
  For i=1 To n

```
    DTW[i, 0] = ∞
  end
  DTW[0, 0] = 0
  For i=1 To n
    For j=1 To m
      Cost = Calculate Distance(S[1..10, i], T[1..10, j])
      DTW[i, j] = Cost + Find Minimum(DTW[i-1, j], DTW[i, j-1], DTW[i-1, j-1])
    end
  end
  Return DTW[n, m]
```

2.6 Artificial neural networks

A Neural Network (NN) is a computer software that simulates a simple model of neural cells in humans. The purpose of this simulation is to acquire the intelligent features of these cells.

Backpropagation networks are a popular type of network that can be trained to recognize different patterns including images, signal, and text. Backpropagation networks have been used for this speech recognition system.

2.6.1 Sigmoid function

The function as Equation 11 is called a Sigmoid function. The coefficient a is a real number constant. In NN applications, a is usually chosen between 0.5 and 2. As a starting point, we can use a=1 and modify it later when we are fine-tuning the network. Note that $s(0) = 0.5$, $s(\infty) = 1$, $s(-\infty) = 0$ (The symbol ∞ means infinity).

$$s(x) = \frac{1}{(1 + e - ax)} \tag{11}$$

The Sigmoid function will convert values less than 0.5 to 0, and values greater than 0.5 to 1. The Sigmoid function is used on the output of neurons.

In NNs, a neuron is a model of a neural cell in humans. This model is simplistic, but as it turned out, is very practical. The neuron has been thought as a program or a class that has one or more inputs and produces one output. The inputs simulate the signals that a neuron gets, while the output simulates the signal which the neuron generates. The output is calculated by multiplying each input by a different number which is called weight, adding them all together, then scaling the total to a number between 0 and 1.

Figure 4 shows a simple neuron with:

a. Three hundred inputs $[x_1, x_2, x_3, ..., x_{300}]$. The input values are usually scaled to values between 0 and 1.
b. 300 input weights $[w_1, w_2, w_3, ..., w_{300}]$. The weights are real numbers that usually are initialized to some random numbers. The weights are variables of type real. We can initialize to a random number between 0 and 1.
c. One output z. A neuron has only one output. Its value is between 0 and 1, it can be scaled to the full range of actual values.

$$d = (x_1 * w_1) + (x_2 * w_2) + (x_3 * w_3) + ... + (x_{300} * w_{300}) \tag{12}$$

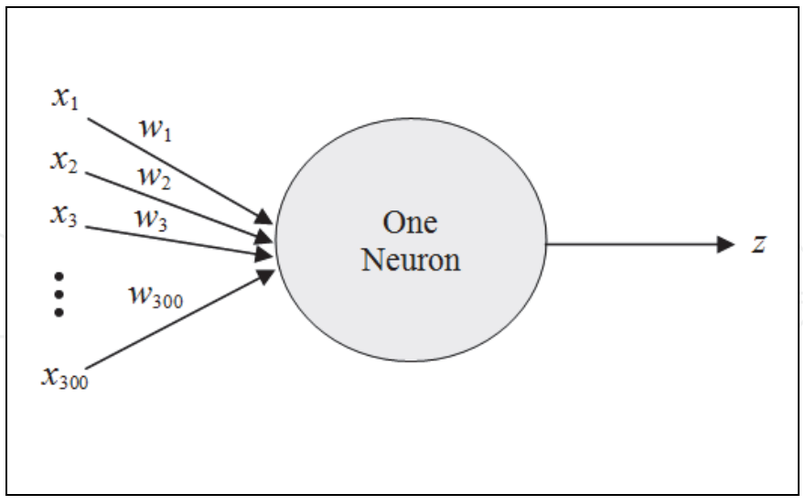

Fig. 4. One neuron structure

In a more general manner, for n number of inputs, d is defined as Equation 13.

$$d = \sum_{i=1}^{n} x_i * w_i \tag{13}$$

Let θ be a real number which is known as a threshold. Experiments have shown that best values for θ are between 0.25 and 1. θ is just a variable of type real that is initialized to any number between 0.25 and 1.

$$z = s(d + \theta) \tag{14}$$

In Equation 14, the output z is the result of applying the sigmoid function on $(d + \theta)$. In NN applications, the challenge is to find the right values for the weights and the threshold.

2.6.2 Backpropagation

The structure of the system is shown in Figure 5. This NN consists of four layers: Input layer with 300 neurons, first hidden layer with 30 neurons, second hidden layer with 10 neurons and output layer with one neuron.

The output of a neuron in a layer goes to all neurons in the following layer. Each neuron has its own input weights. The weights for the input layer are assumed to be 1 for each input. In other words, input values are not changed. The output of the NN is reached by applying input values to the input layer, passing the output of each neuron to the following layer as input. The Backpropagation NN must have at least an input layer and an output layer. It could have zero or more hidden layers.

The number of neurons in the input layer depends on the number of possible inputs while the number of neurons in the output layer depends on the number of desired outputs. In general, the addition of a hidden layer could allow the network to learn more complex patterns, but at the same time decreases its performance.

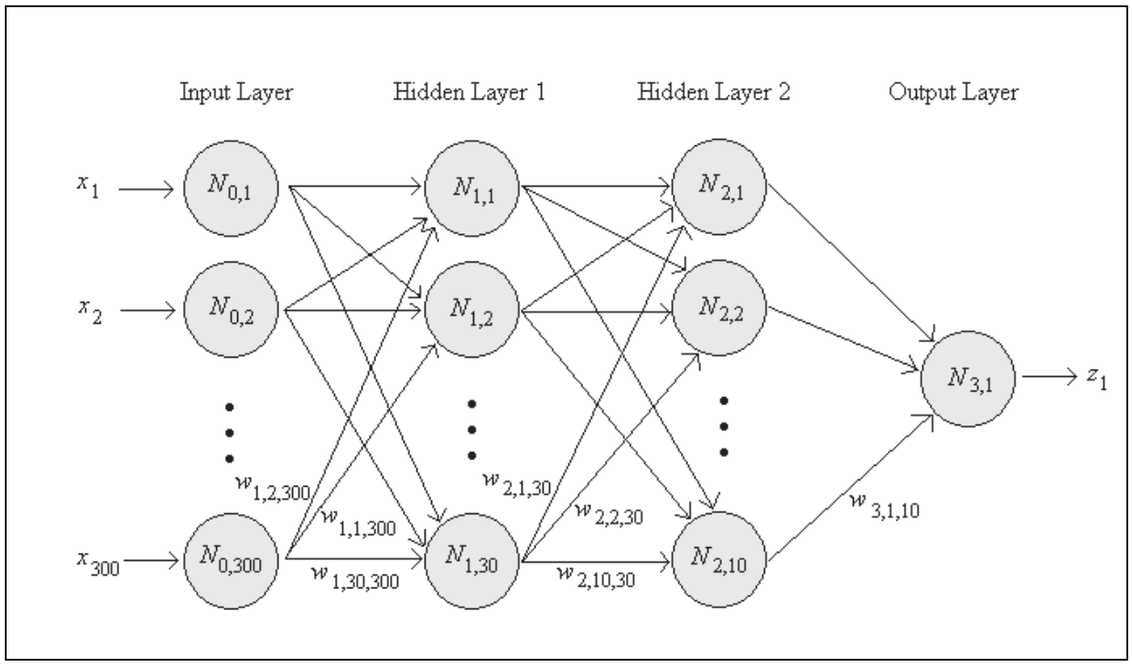

Fig. 5. Backpropagation network of the system

2.6.3 Supervised learning

The Backpropagation NN works supervised training. The training can be summarized as the following algorithm.

Step 1. Start by initializing the input weights for all neurons to some random numbers between 0 and 1.

Step 2. Apply input to the network.

Step 3. Calculate the output.

Step 4. Compare the resulting output with the desired output for the given input. This is called the error.

Step 5. Modify the weights and threshold θ for all neurons using the error.

Step 6. Repeat the process until the error reaches an acceptable value (the error, 0.006), which means that the NN was trained successfully, or if we reach a maximum count of iterations, which means that the NN training was not successful.

The challenge is to find a good algorithm for updating the weights and thresholds in each iteration (Step 5) to minimize the error.

Changing weights and threshold for neurons in the output layer is different from hidden layers. For the input layer, weights remain constant at 1 for each input neuron weight.

For the training operation, the followings are defined.

The Learning Rate, λ : A real number constant, 0.02 for the system.

The change, Δ : For example Δx is the change in x.

2.6.3.1 Output layer training

Let z be the output of an output layer neuron. Let y be the desired output for the same neuron, it should be scaled to a value between 0 and 1. This is the ideal output which we like to get when applying a given set of input. Then the error, e, will be as Equation 15.

$$e = z * (1 - z) * (y - z) \tag{15}$$

$$\Delta\theta = \lambda * e \tag{16}$$

$$\Delta w_i = \Delta\theta * x_i \tag{17}$$

$\Delta\theta$ represents the change in θ. Δw_i is the change in weight *i* of the neuron. In other words, for each output neuron, calculate its error *e*, and then modify its threshold and weights using Equation 15, 16 and 17.

2.6.3.2 Hidden layer training

Consider a hidden layer neuron as shown in Figure 6. Let z be the output of the hidden layer neuron. Let m_i be the weight at neuron N_i in the layer following the current layer. This is the weight for the input coming from the current hidden layer neuron. Let e_i be the error e at neuron N_i. Let r be the number of neurons in the layer following the current layer. (In Figure 6, $r = 3$).

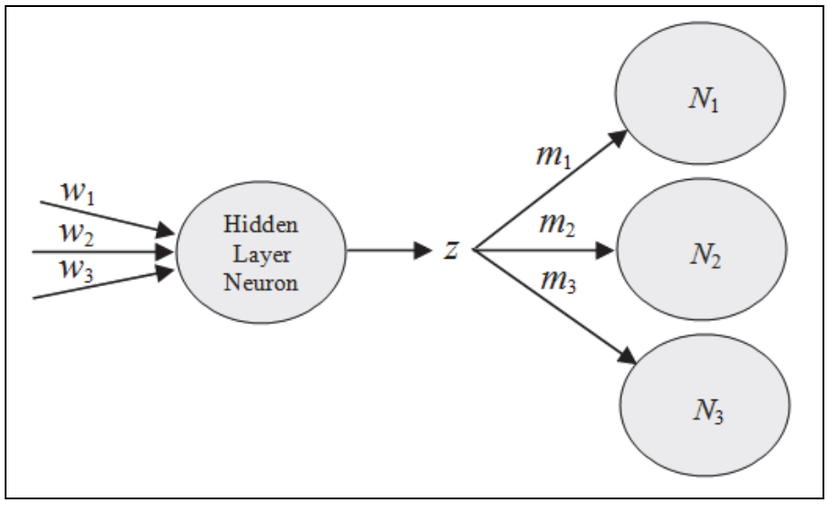

Fig. 6. Hidden layer training

$$g = \sum_{i=1}^{r} m_i * e_i \tag{18}$$

$$e = z * (1 - z) * g \tag{19}$$

$$\Delta\theta = \lambda * e \tag{20}$$

$$\Delta w_i = \Delta\theta * x_i \tag{21}$$

e is the error at the hidden layer neuron. $\Delta\theta$ is the change in θ. Δw_i is the change in weight *i*. In calculating g, we use the weight m_i and error e_i from the following layer, which means that the error and weights in this following layer should have already been

calculated. This implies that during a training iteration of a Backpropagation NN, we start modifying the weights at the output layer, and then we proceed backwards on the hidden layers one by one until we reach the input layer. It is the method of proceeding backwards which gives this network its name Backward Propagation.

2.7 The postprocessing of the system

After the syllables of the word utterance are recognized using the speech recognition method, and the most similar 10 syllables are put in order, the recognized syllables are concatenated and generated the recognized word. We can find the most similar words in order by concatenation of the most similar syllables. From the uppermost recognized words, it can be determined whether or not the word is Turkish. If the word is Turkish, it is the recognized word of the system. If these words are not Turkish, the system does not recognize any word.

For example, as shown in Table 1, the recognized syllables are ordered. Hence, the most similar syllables as "kı", "tap" and "lik" have been found. These syllables are concatenated, and the most similar word as "kıtaplik" is constructed. But, the system decides that the word is not Turkish word. Then, the next most similar word is concatenated, and it is determined whether or not the word is Turkish. This process is continued until a Turkish word is found in these concatenated words. In this example, the word "kitaplık" which is generated from the syllables "ki", "tap" and "lık" is detected by the system. Therefore, it is the recognized word using the postprocessing.

The order of the most similar syllables	Recognized Syllables		
	"ki"	**"tap"**	**"lık"**
1.	kı	tap	lik
2.	ki	tap	lak
3.	ki	tep	lık
4.	ki	ta	lik
5.	kı	ta	lık

Table 1. The most similar syllables

3. Detecting misspelled words using syllable n-gram frequencies

To detect misspelled words in a text is an old problem. Today, most of word processors include some sort of misspelled word detection. Misspelled word detection is worthy in the area of cryptology, data compression, speech synthesis, speech recognition and optical character recognition (Barari & QasemiZadeh, 2005; Deorowicz & Ciura, 2005; Kang & Woo, 2001; Tong & Evans, 1996). The traditional way of detecting misspelled words is to use a word list, usually also containing some grammatical information, and to look up every word in the word list (Kukich, 1992) from dictionary.

The main disadvantage of this approach is that if the dictionary is not large enough, the algorithm will report some of correct words as misspelled, because they are not included in the dictionary. For most natural languages, the size of dictionary needed is too large to fit in the working memory of an ordinary computer. In Turkish this is a big problem, because Turkish is an agglutinative language and too many new words can be constructed by adding suffixes.

To overcome this difficulties, a new approach has been proposed for detecting misspelled words in Turkish text. For that, Turkish syllable *n*-gram frequencies which are generated from several Turkish corpora have been used. From the corpora, syllable monogram, bigram and trigram frequencies have been extracted using TASA (Aşlıyan & Günel, 2005). These *n*-gram frequencies have been used for calculating a word probability distribution. After that the system has decided whether a word is misspelled or not. This approach does not need any word list.

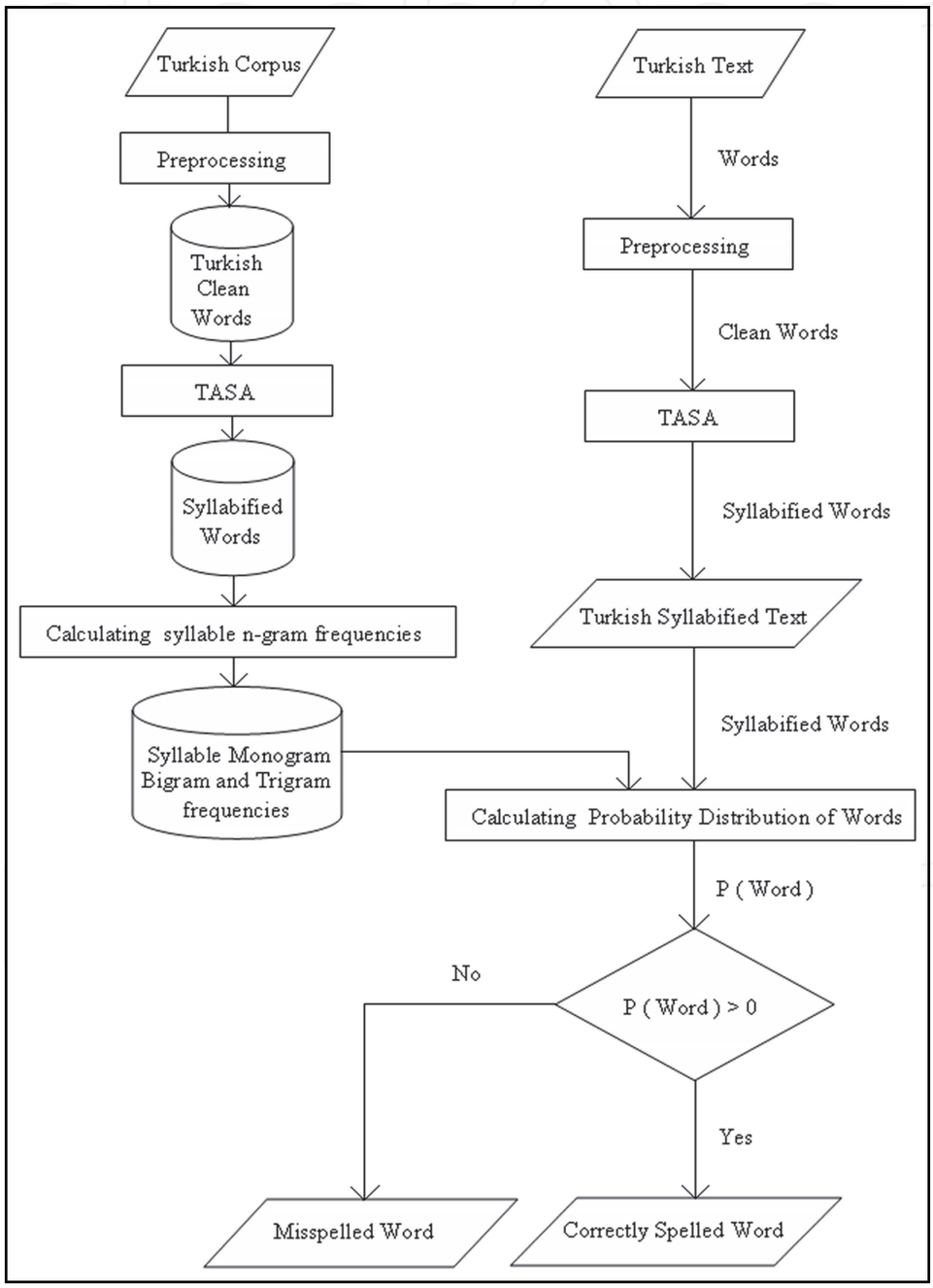

Fig. 7. The system architecture

3.1 System architecture

The system consists of three main components. First component is preprocessing which cleans a text. Second component is TASA, and third component is calculating probability distribution of words. As shown in Figure 7, the system takes words in Turkish text as input and gives the result for each word as "Misspelled Word" or "Correctly Spelled Word".

In preprocessing component of the system, punctuation marks are cleaned. All letters in the text are converted to lower case. Blank characters between two successive words are limited with only one blank character.

In second component, TASA takes the Turkish clean text as an input and gives the Turkish syllabified text. The system divides words into syllables putting the dash character between two syllables. For example, the word "kitaplık" in Turkish text is converted into the syllabified word "ki-tap-lık" in Turkish syllabified text.

In third component, the probability distribution is calculated for each syllabified word. The system uses syllable monogram, bigram and trigram frequencies to find this probability distribution.

3.2 Calculation of the probability distribution of words

An *n*-gram is a sub-sequence of *n* items from a given sequence. *n*-grams are used in various areas of statistical natural language processing and genetic sequence analysis. The items in question can be letters, syllables, words according to the application.

An *n*-gram of size 1 is a "monogram"; size 2 is a "bigram"; size 3 is a "trigram"; and size 4 or more is simply called an "*n*-gram" or "(*n*−1)-order Markov model" (Zhuang et al., 2004).

An *n*-gram model predicts x_i based on $x_{i-1}, x_{i-2}, x_{i-3}, ..., x_{i-n}$. When used for language modelling independence assumptions are made so that each word depends only on the last *n* words. This Markov model is used as an approximation of the true underlying language. This assumption is important because it massively simplifies the problem of learning the language model from data.

Suppose that a word *W* in Turkish syllabified text consists of the syllable sequence $s_1, s_2, s_3, ..., s_t$. This word has *t* syllables. To obtain the *n*-gram probability distribution (Jurafsky & Martin, 2000) of the word W, we have used in Equation 22.

$$P(W) = P(s_1, s_2, ..., s_t) = \prod_{i=1}^{t} P(s_i \mid s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}) \tag{22}$$

In *n*-gram model, the parameter $P(s_i \mid s_{i-n+1}, s_{i-n+2}, ..., s_{i-1})$ in Equation 22 can be estimated with Maximum Likelihood Estimation (MLE) (Aşlıyan & Günel, 2005) technique as shown in Equation 23.

$$P(s_i \mid s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}) = \frac{C(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}, s_i)}{C(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1})} \tag{23}$$

So, it is concluded as Equation 24.

$$P(W) = P(s_1, s_2, ..., s_t) = \prod_{i=1}^{t} \frac{C(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1}, s_i)}{C(s_{i-n+1}, s_{i-n+2}, ..., s_{i-1})} \tag{24}$$

In Equation 23 and Equation 24, $C(s_{i-n+1},s_{i-n+2},...,s_{i-1},s_i)$ is the frequency of the syllable sequence $s_{i-n+1},s_{i-n+2},...,s_{i-1},s_i$. Furthermore, $C(s_{i-n+1},s_{i-n+2},...,s_{i-1})$ is the frequency of the syllable sequence $s_{i-n+1},s_{i-n+2},...,s_{i-1}$. The frequencies of these syllable sequences can be calculated from some Turkish corpora.

For bigram and trigram models, probability distribution $P(W)$ can be computed as shown in Equation 25 and Equation 26 respectively.

$$P(W)=P(s_1,s_2,...,s_t)=\prod_{i=1}^{t}P(s_i\mid s_{i-1})=\prod_{i=1}^{t}\frac{C(s_{i-1},s_i)}{C(s_{i-1})} \tag{25}$$

$$P(W)=P(s_1,s_2,...,s_t)=\prod_{i=1}^{t}P(s_i\mid s_{i-2},s_{i-1})=\prod_{i=1}^{t}\frac{C(s_{i-2},s_{i-1},s_i)}{C(s_{i-2},s_{i-1})} \tag{26}$$

For example, according to bigram model it can be calculated the probability distribution of a word in Turkish text. Assume that we have a text which includes some words as "... Bu gün okulda, şenlik var...". This text is converted to syllabified text as "... Bu gün o-kul-da, şen-lik var...". Syllables in the words are delimited with dash character. Assume that the word *W*="okulda" in the text is taken for computing its probability distribution and *W* can be written as the syllable sequence $W=s_1,s_2,s_3=$"o", "kul", "da". Here, $s_1=$"o", $s_2=$"kul", $s_3=$"da". Blank character is accepted as a syllable. This syllable is called as λ. So, assume that syllable monogram frequencies are *C*("λ")=0.003, *C*("o")=0.002, *C*("kul")=0.004 and syllable bigram frequencies are *C*("λ","o")=0.0001, *C*("o","kul")=0.0002, *C*("kul","da")=0.0003. *P*("okulda") has been calculated using bigram model. It can be found that the probability distribution of the word "okulda" is 0.0002475 as shown in Equation 27.

$$\begin{aligned}P(W)&=P(\text{"okulda"})=P(s_1,s_2,s_3)=P(\text{"o","kul","da"})\\&=\prod_{i=1}^{3}P(s_i\mid s_{i-1})=\prod_{i=1}^{3}\frac{C(s_{i-1}\mid s_i)}{C(s_{i-1})}\\&=\left(\frac{C(\text{"}\lambda\text{","o"})}{C(\text{"}\lambda\text{"})}\right)\left(\frac{C(\text{"o","kul"})}{C(\text{"o"})}\right)\left(\frac{C(\text{"kul","da"})}{C(\text{"kul"})}\right)\end{aligned} \tag{27}$$

3.3 Testing and evaluation of the system

Two systems have been designed and implemented to detect misspelled words in Turkish text. One uses monogram and bigram frequencies. The size of monogram database is 41 kilobytes and the monogram database consists of 4141 different syllables. The sizes of bigram and trigram databases are 570 and 2858 kilobytes respectively. While the bigram database includes 46684 syllable pairs, the trigram database consists of 183529 ternary syllables. The other uses bigram and trigram frequencies. These two systems have been tested for two Turkish texts. One is correctly spelled text which includes 685 correctly spelled words. The other is misspelled text which has 685 misspelled words. These two texts have same words. Namely, misspelled words are generated with putting errors on the correctly spelled words. These error types are substitution, deletion, insertion, transposition and split word errors. The system takes correctly spelled and misspelled texts as input and

gives the results for each word as "correctly spelled word" or "misspelled word". Probability distributions are calculated for each word. If the probability distribution of a word is equal to zero, system decides that the word is misspelled. If it is greater than zero, system decides that the word is correctly spelled.

Firstly, the system on the correctly spelled text has been tested using monogram and bigram frequencies. The system determines 602 correctly spelled words from the correctly spelled text, so the words are correctly recognized with 88% success rate. Also, 589 misspelled words within the misspelled text are decided successfully by the system. Namely, the system which is tested on the misspelled text correctly recognized the words with 86% success rate.

Then the system on the correctly spelled text has been tested using bigram and trigram frequencies. The system determines 671 of 685 correctly spelled words from the correctly spelled text. The success rate on correctly recognition of the words is 98%. Furthermore, 664 of 685 misspelled words within the misspelled text are decided successfully by the system. Thus, the system which is tested on the misspelled text correctly recognized the words with 97% success rate. The system can analyze 75000 words per second.

4. Experimental results

In this section, the extperimental results of the developed syllable based speech recognition have been given according to DTW and ANN approach, and the success rates of the systems have been compared.

The most widely used evaluation measure for speech recognition performance is Word Error Rate (WER) (Hunt, 1990; McCowan et al., 2005). The general difficulty of measuring the performance lies on the fact that the recognized word sequence can have different length from the reference word sequence. The WER is derived from the Levenshtein distance, working at word level instead of character.

This problem is solved by first aligning the recognized word sequence with the reference sequence using dynamic string alignment. The word error rate can then be computed as in Equation 28.

$$WER = 100\left(\frac{E}{N}\right) \tag{28}$$

where E is the number of wrongly detected words, and N is the number of words in the reference set.

In Figure 8, the WER results are given for dynamic time warping. If we evaluate the system, we can say that the best result for DTW is obtained with the mfcc feature. It is followed by rasta feature. The system accuracy rate is increased with postprocessing operation about 9% using DTW.

In Figure 9, the WER results are given for artificial neural network. If we evaluate the system, it can be said that the best result for ANN is obtained with the mfcc feature. The system accuracy rate is increased with postprocessing operation about 15% using ANN.

According to the results of WER of the system, the most successful feature and speech recognition method are mfcc and DTW (5.8% WER) respectively. The postprocessing improves approximately 14% of the system accuracy.

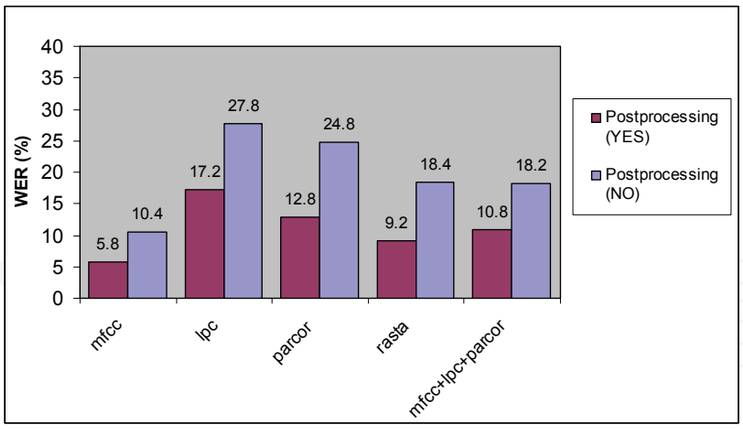

Fig. 8. WER results of system using DTW

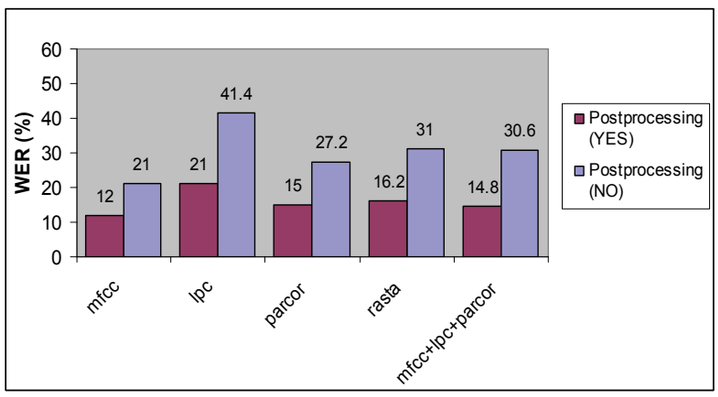

Fig. 9. WER results of system using ANN

DTW does not need training operation. It uses the extracted features. The best feature is mfcc because its extraction time duration is the lowest. ANN needs training, and it constructs a model for each syllable in words using training. As shown in Table 2, the average training time for ANN is about 1102.5 seconds.
Table 3 displays average testing time duration. Average testing time duration of ANN (7.4 seconds) is quite shorter than that of DTW.

Methods	Training Time (Seconds)
ANN	1102.5

Table 2. Average training time for one syllable

Methods	Testing Time (Seconds)
DTW	57.3
ANN	7.4

Table 3. Average testing time for one syllable

5. Conclusion

In this study, syllable based isolated word Turkish speech recognition systems using the speech recognition methods as DTW and ANN have been designed and implemented. These speaker dependent systems use the features as mfcc, lpc, parcor, cepstrum, rasta and the mixture of mfcc, lpc and parcor. Syllable models of the words in the dictionary are constructed syllable databases to compare the word utterence. The system firstly recognizes the syllables of the word utterence. Recognized word is found by the concatenation of the recognized syllables.

To use in postprocessing stage of the system, firstly, TASA have been designed and implemented. TASA's correct spelling rate is about 100%. Then, Turkish syllable *n*-gram frequencies for some Turkish corpora have been calculated. In addition, using syllable *n*-gram frequencies, a system which decides whether or not a word is misspelled in Turkish text has been developed. The system takes words as inputs. The system produces two results for each word: "Correctly spelled word" or "Misspelled word". According to the system designed with bigram and trigram frequencies, the success rate is 97% for the misspelled words, and 98% for the correctly spelled words.

In postprocessing operation, after the recognized word is constructed by concatenating of the recognized syllables, the system decides whether it is Turkish word or not. If the word is Turkish word, then it is new recognized word. This postprocessing increases the accuracy rate of the system approximately 14%.

After testing of the middle scaled speech recognition system, the most successful method is DTW whose word error rate is about 5.8%. It can be said that the best feature for the speech recognition is mel frequency cepstral coefficients.

For future work, Support Vector Machine and Hidden Markov Model can be applied for syllable based speech recognition, and we can compare among the results obtained using these speech recognition methods.

6. Acknowledgment

The author would like to thank the Scientific Research Unit (BAP) of Adnan Menderes University.

7. References

Arısoy, E. & Dutağacı, H. (2006). A unified language model for large vocabulary continuous speech recognition of Turkish. *Signal Processing*, Vol.86, No.10, pp. 2844-2862

Artuner, H. (1994). *The design and implementation of a Turkish speech phoneme clustering system*, PhD Thesis, Hacettepe University, Ankara, Turkey

Aşlıyan, R. & Günel, K. (2005). Design and implementation for extracting Turkish syllables and analysing Turkish syllables, *INISTA (International Symposium on Innovations in Inttelligent Systems and Applications)*, pp. 170-173, ISBN 975-461-400-8, İstanbul, Turkey, 15-18 June 2005

Aşlıyan, R.; Günel, K. & Yakhno, T. (2007). Detecting misspelled words in Turkish text using syllable n-gram frequencies. *Lecture Notes in Computer Science (LNCS)*, Vol.4815, pp. 553-559

Aşlıyan, R. (December 2010). Speech Recognition System Databases, 15.01.2011, Available from http://web.adu.edu.tr/akademik/rasliyan/speech/sdw.html

Avcı, E. (2007). An automatic system for Turkish word recognition using discrete wavelet neural network based on adaptive entropy. *Arabian Journal for Science and Engineering*, Vol.32, pp. 239-250

Barari, L. & QasemiZadeh, B. (2005). CloniZER spell checker adaptive language independent spell checker, *AIML 05 Conference CICC*, pp. 19-21, Cairo, Egypt

Blanz, V.; Schölkopf, B., Bulthoff, H., Burges, C., Vapnik, V. N. & Vetter, T. (1996). Comparison of view-based object recognition algorithms using realistic 3D models, *Lecture Notes in Computer Science (LNCS)*, Vol.1112, pp. 251-256

Cortes, C. & Vapnik, V. N. (1995). Support vector network. *Machine Learning*, Vol.20, pp. 1-25

Davis, K.; Biddulph, R., & Balashek, S. (1952). Automatic recognition of spoken digits. *J. Acous. Soc. Ame.*, Vol.24, pp. 3-50

Deorowicz, S. & Ciura M. G. (2005). Correcting spelling errors by modelling their causes. *International Journal of Applied Mathematics and Computer Science*, Vol.15, No.2, pp. 275-285

Dreyfus-Graf, J. (1952). Letyposonographe phonetique ou phonetographe. *Bulletin Technique Des PTT Suisses*, Vol.12, pp. 363-379

Dudley, H. & Balashek, S. (1958). Automatic recognition of phonetic patterns in speech. *J. Acoustic Soc. Am.*, Vol.30, pp. 721-733

Ferguson, J. (1980). *Hidden Markov Models for speech*, IDA-CRD, Princeton, New Jersey

Forgie, J. & Forgie, C. (1959). Results obtained from a vowel recognition computer program. *J. Acoust. Soc. Ame.*, Vol.31, pp. 1480-1489

Fornés A.; Lladós J., Sánchez G. & Karatzas D. (2010). Rotation invariant hand-drawn symbol recognition based on a dynamic time warping model. *International Journal on Document Analysis and Recognition*, Vol.13, No.3 pp. 229-241

Haton, J. P. & Pierrel, J. M. (1976). Organization and operation of a connected speech understanding system at lexical, syntactical and semantical levels. In : *ICASSP*, pp. 430-433

Hunt, M. J. (1990). Figures of merit for assessing connected word recognisers. *Speech Communication*, Vol.9, pp. 229-236

Jakobson, R.; Fant, G., & Halle, M. (1952). *Preliminaries to speech analysis*, MIT Press, ISBN 0-262-60001-3 Cambridge, MA

Junqua, J. C.; Mak, B. & Reaves, B. (1997). A robust algorithm for word boundary detection in the presence of noise. *IEEE Transactions on Speech and Audio Processing*, Vol.2, No.3, pp. 406-412

Jurafsky, D. & Martin, J. H. (2000). *Speech and language processing*, Prentice Hall, ISBN 0-13-095069-6 New Jersey

Kang, S. S. & Woo, C. W. (2001). Automatic segmentation of words using syllable bigram statistics, *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium*, pp. 729-732, Tokyo, Japan

Karaca, N. (1999). *Realization of a Turkish isolated word speech recognition system under noisy environments*, PhD Thesis, Hacettepe University, Ankara, Turkey

Koç, A. (2002). *Acoustic feature analysis for robust speech recognition*, MSc Thesis, Boğaziçi University, İstanbul, Turkey

Kukich, K. (1992). Techniques for automatically correcting words in text. *ACM Computing Surveys*, Vol.24, No.4, pp. 377-439

Lesser, V.; Fennell, R., Erman, L., & Reddy, D. (1975). Organization of the HEARSAY II speech understanding system. *IEEE Trans. ASSP*, Vol.23, No.1, pp. 11-23

Lippmann, R. (1987). An introduction to computing with neural nets. *IEEE Trans. ASSP Magazine*, Vol.4, No.2, pp. 4-22

Lowerre, B. (1976). *The harpy speech recognition system*, Technical Report, Carnegie Mellon University

McCowan, I.; Moore, D., Dines, J., Gatica-Perez, D., Flynn, M., Wellner, P., et al. (2005). *On the use of information retrieval measure for speech recognition evaluation*, IDIAP, IDIAP-RR 73

Meral, O. (1996). *Speech recognition based on pattern comparison techniques*, MSc Thesis, İstanbul Technical University, İstanbul

Mercier, G. (1977). A multipurpose speech understanding system, In: *ICASSP*, pp. 815-818

Myers, C.; Rabiner, L. & Rosenberg, A. (1980). Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol.6, pp. 623-635

Nagata, K.; Kato, Y., & Chiba, S. (1963). Spoken digit recognizer for Japanese language. *NEC Res. Develop.*, Vol.6, No.2

Olson, H. & Bellar, H. (1956). Phonetic typewriter. *J. Accous. Soc. Ame.*, Vol.2, pp. 1072-1081

Osuna, E.; Freud, R. & Girosi, F. (1997). Training support vector machines: An applications to face detection, In : *CVPR97*, pp. 130-136

Özkan, Ö. (1997). *Implementation of speech recognition for connected numerals*, MSc Thesis, Middle East Technical University, Ankara, Turkey

Rabiner, L. & Sambur, M. R. (1975). An algorithm for determining the end-points of isolated utterances. *The Bell System Technical Journal*, Vol.54, No.2, pp. 297-315

Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proc. IEEE*, Vol.77, No.2, pp. 257-286

Reddy, D. (1966). An approach to computer speech recognition by direct analysis of the speech wave, Technical Report, Stanford University

Sakai, T. & Doshita, S. (1962). The phonetic typewriter. In: *IFIP Congress*, pp. 445-449

Salor, Ö & Pellom, B. L. (2007). Turkish speech corpora and recognition tools developed by porting SONIC: Towards multilingual speech recognition. *Computer Speech and Language*, Vol.21, No.4, pp. 580-583

Sankoff, D. & Kruskal, J. (1999). *Time Warps, String Edits, and Macromolecules–The Theory and Practice of Sequence Comparison*, The David Hume Series, Stanford, CA

Savoji, M. H. (1989). End-pointing of speech signals. *Speech Communication*, Vol.8, No.1, pp. 46-60

Slutsker, G. (1968). Nelinejnyp method analiza recevych signalov. *Trudy NIIR*, Vol.2, pp. 76-82

Tong, X. & Evans, D. A. (1996). A statistical approach to automatic OCR error correction in context, *Proceedings of the Fourth Workshop on Very Large Corpora*, pp. 88-100, Copenhagen, Denmark

Vintsyuk, T. (1968). Speech discrimination by dynamic programming. *Cybernetics (Kibernetika)*, Vol.4, No.1, pp. 81-88

Wolf, J. & Woods, W. (1977). The HWIM speech understanding system. In: *ICASSP*, pp. 784-787

Yılmaz, C. (1999). A large vocabulary speech recognition system for Turkish, M. S. Thesis, Bilkent University, Ankara, Turkey

Zhuang, L.; Bao, T., Zhu, X.,Wang, C. & Naoi, S. (2004). A chinese OCR spelling check appoarch based on statistical language models, *IEEE International Conference on Systems, Man and Cybernetic*, pp. 4727-4732

Speech Technologies

Edited by Prof. Ivo Ipsic

ISBN 978-953-307-996-7
Hard cover, 432 pages
Publisher InTech
Published online 23, June, 2011
Published in print edition June, 2011

This book addresses different aspects of the research field and a wide range of topics in speech signal processing, speech recognition and language processing. The chapters are divided in three different sections: Speech Signal Modeling, Speech Recognition and Applications. The chapters in the first section cover some essential topics in speech signal processing used for building speech recognition as well as for speech synthesis systems: speech feature enhancement, speech feature vector dimensionality reduction, segmentation of speech frames into phonetic segments. The chapters of the second part cover speech recognition methods and techniques used to read speech from various speech databases and broadcast news recognition for English and non-English languages. The third section of the book presents various speech technology applications used for body conducted speech recognition, hearing impairment, multimodal interfaces and facial expression recognition.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen