Chapter 4

Speech Recognition using Features based on Wavelet Transform

In the chapter 3, we have discussed the models and their performance for speech recognition of Gujarati words using the feature extraction method: mel-frequency cepstral coefficients (MFCC). Now, in this chapter, we will discuss models for the speech recognition of Gujarati words and Gujarati sentences using the discrete wavelet transform based feature extraction method: mel-frequency discrete wavelet coefficients (MFDWC). The wavelets are good for non-stationary signals like speech, and they give simultaneous details in time and frequency domains. Due to this fact, we can expect better recognition accuracy in the models based on mel-frequency discrete wavelet coefficients as compared to the ones that use mel-frequency cepstral coefficients, as explained in chapter 3.

The models explained in this chapter are divided into two parts. In the first part, a dataset for isolated words is considered. For the classification of the isolated words, we have used techniques like dynamic time warping, multilayered perceptrons, radial basis function networks, and hidden Markov models. In the second part, a dataset for continuous sentences is considered, and for further processing, the words are extracted from sentences automatically using the signal processing technique of short-term autocorrelation. For continuous sentence recognition, we have used multilayered perceptrons and hidden Markov models. Subsequent sections discuss the model details and results.

First, we discuss the initial steps of the speech recognition process, namely feature extraction and making the lengths of the feature vector equal.

4.1 Feature Extraction using MFDWC

In this chapter, a different feature extraction technique is used. Let us try to understand each step of this method by using an example. Consider a raw speech signal containing the word "Ek", meaning digit 'One', spoken in the Gujarati language as shown in the Figure 4.1.

The preprocessing phase before feature extraction is similar to that explained in chapter 3, which is detailed below. We apply the first step of pre-emphasising on this speech signal with $\alpha = 0.97$ to get a pre-emphasised speech signal as shown in the Figure 4.2. Due to this step, the amount of energy at high frequencies increases. As a result, information from higher formants is more available to the recognition process, which makes the signal spectrally flat. This makes it less susceptible to finite precision effects.

The next step is to divide a signal into various overlapping frames. Considering that the framing is done with a frame width of 256 samples and overlapping of 100 samples, the resulting thirteenth frame would look like Figure 4.3. The sampling rate considered was 16,000 samples per second. So, the window shown in the Figure 4.3 corresponds to the part of speech of about 16 milliseconds. As we can see in the Figure 4.3, there are fewer variations within this window. This helps us analyse signals frame-by-frame.

After that, we apply the hamming window with $\beta = 0.46$. The resultant speech signal is shown in the Figure 4.4. This step is useful to remove discontinuities caused by the previous step while framing the signal. It shrinks the sample values of the signal to zero towards both boundaries of the window, which can be observed in the Figure 4.4.

Then Figure 4.5 shows the result of applying the Fourier transform to the windowed speech signal. The power spectrum is also shown in Figure 4.5. This gives us information about the amount of energy present at various frequency levels. For the given word, the frequency region between 1 Hz and 4 Hz shows some activity in the Figure 4.5.

After that, the frequencies are converted to mel. Then, the filter bank of 20 triangular filters is applied to the power spectrum. They collect energy from each frequency band. Finally, Figure 4.6 represents the result of applying the discrete wavelet transform to the logarithm of the mel-scaled filter bank. For better understanding via visualisation, Figures 4.7 and 4.8 are shown, which represent the Fourier transform, power spectrum, and mel-frequency discrete wavelet coefficients of the entire speech signal. For each frame, the first 10 coefficients are taken as feature vectors.

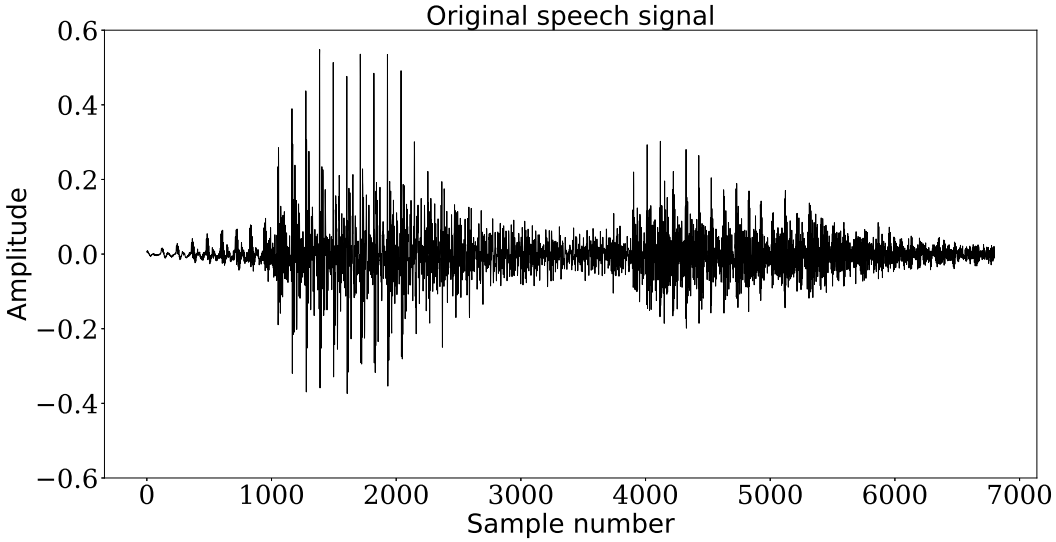

Figure 4.1: Original speech signal consisting of one word

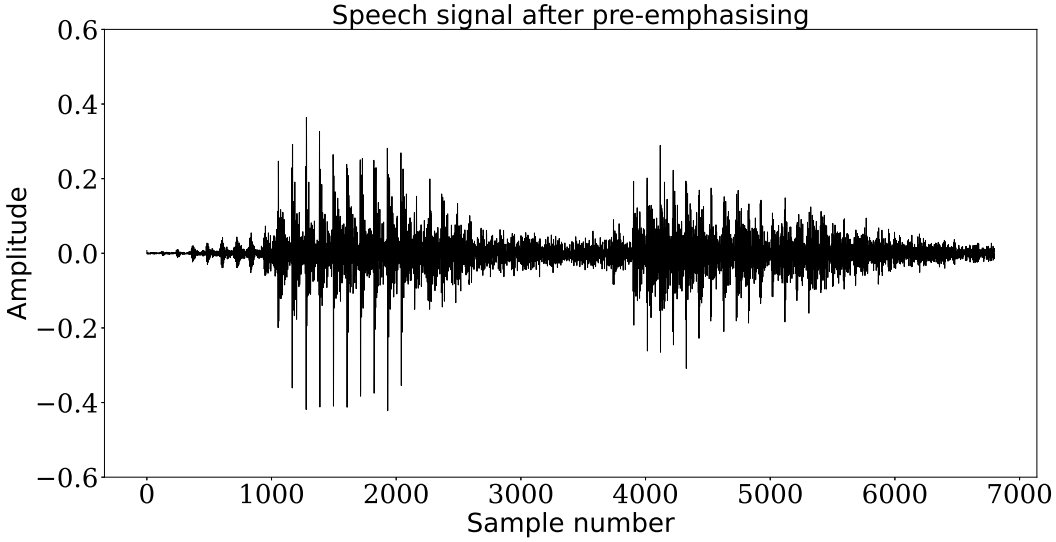

Figure 4.2: Speech signal after applying pre-emphasising with α=0.97

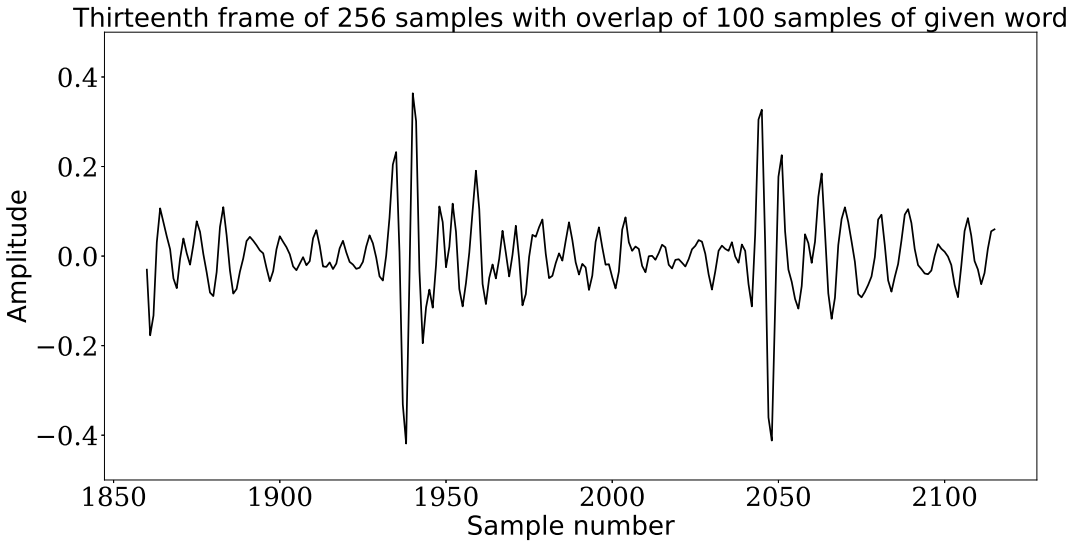

Figure 4.3: Thirteenth frame of the speech signal

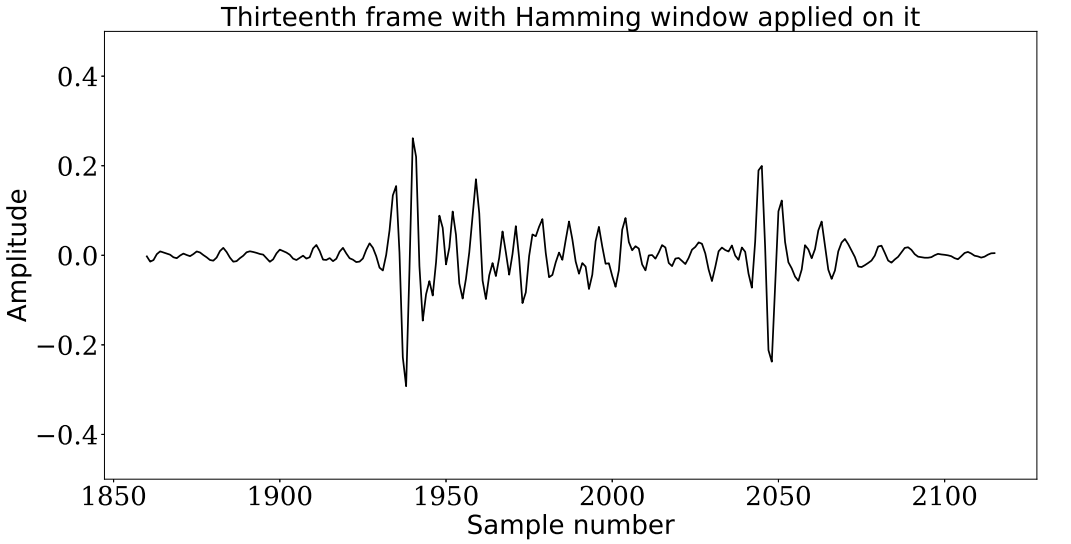

Figure 4.4: Hamming window of the thirteenth frame of the speech signal

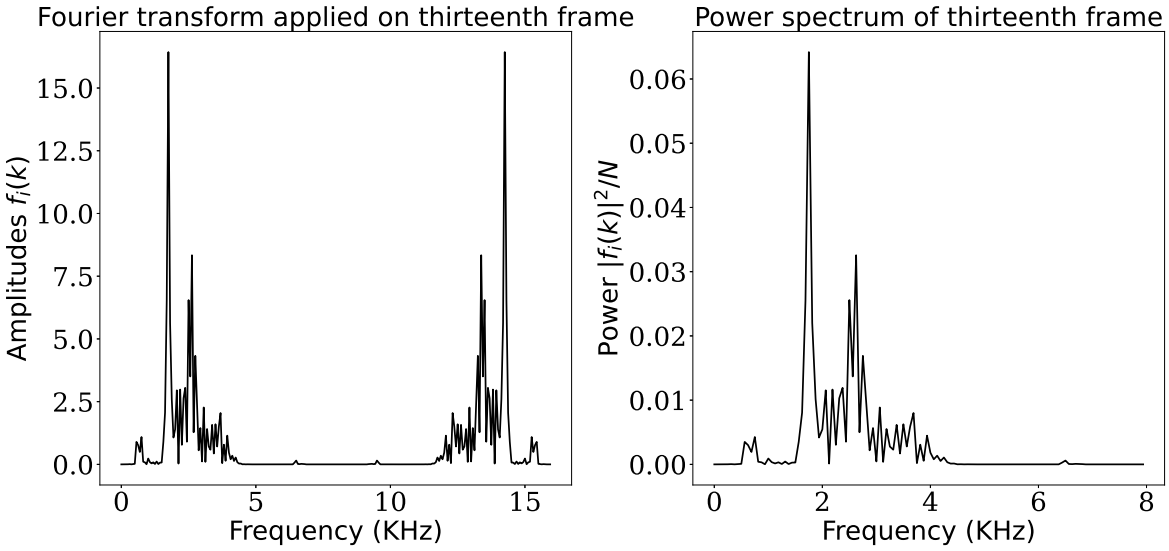

Figure 4.5: Fourier transform and Power spectrum of the thirteenth frame of the speech signal

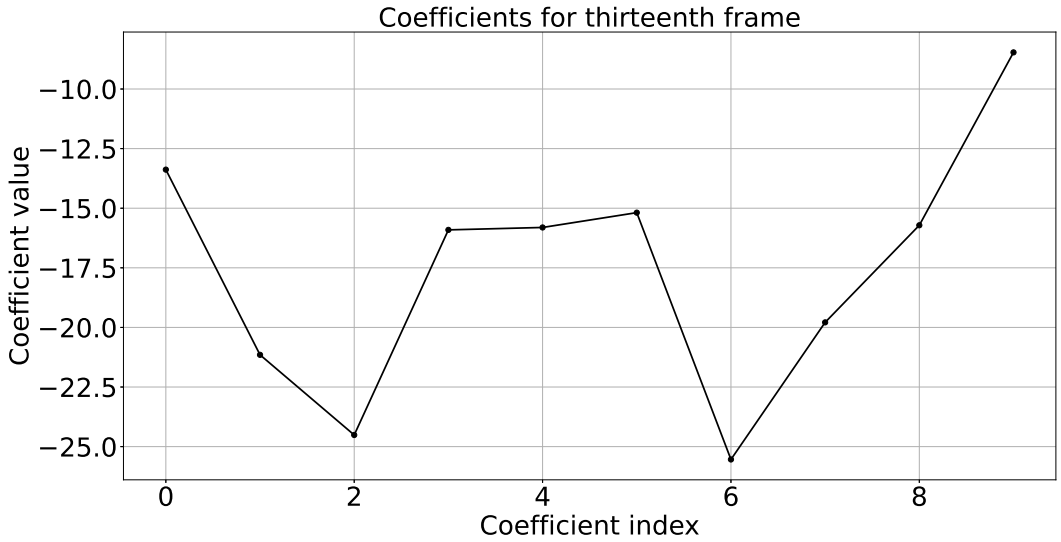

Figure 4.6: Mel-frequency discrete wavelet coefficients of thirteenth frame

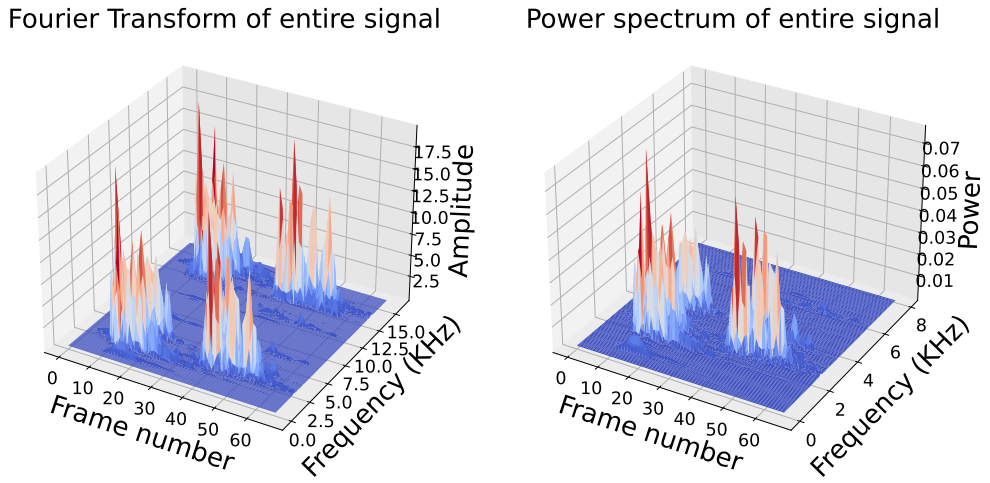

Figure 4.7: Fourier transform and Power spectrum of entire speech signal

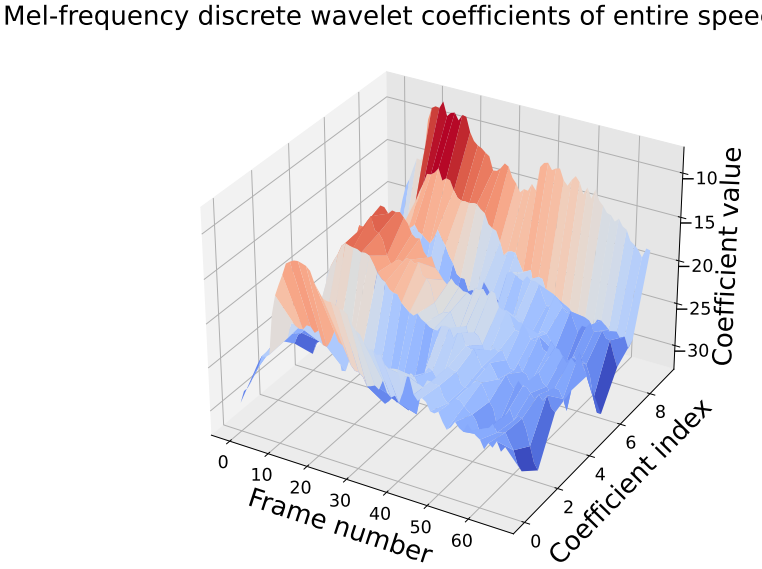

Figure 4.8: Mel-frequency discrete wavelet coefficients of entire speech signal

4.2 Making the Lengths of Feature Vector Equal

For various models in this chapter, the lengths of feature vectors are required to be made equal. It is done by cubic spline interpolation. In this method, we make the lengths of all the feature vectors equal to the median length. This requires shorter feature vectors to be stretched and others to shrink. The feature vectors, whose length is less than the median length, are stretched by including a required number of values in them using cubic spline interpolation. Geometrically, this adds more points to the curve fitted to the data, so it will preserve the geometry of the vector. The feature vectors, whose length is greater than the median length, are shrunk by removing a few values from them using cubic spline interpolation. Geometrically, this removes some points on the curve fitted to the data.

Let us try to understand this by using an example. Suppose we have two arrays, A and B. Array A is longer, and array B is shorter. Suppose we want to shrink array A such that its length is equal to the length of B. We consider the following steps:

1. We use cubic spline interpolation to fit a smooth curve through the data points in array A.
2. Then, the interpolation function is used to estimate the values of array A at positions corresponding to array B.
3. After that, array A is resampled to match the length of array B. The resulting array would contain interpolated values based on the original data points in array A.

These preprocessed feature vectors are used in models for classification using machine learning techniques.

4.3 Models for Recognition of Isolated Words

In this part, we discuss models for the recognition of isolated words. Similar to the subprocess explained in the previous chapter, the speech spoken by various speakers in Gujarati is recorded first. Then, using the open-source software Audacity, the words are cropped near the unvoiced region manually. These words are stored as separate *.wav files. Then, their features are extracted and processed further for classification using various techniques. The details about pre-processing, word extraction, and feature extraction are briefed in the next subsection.

4.3.1 Pre-processing, Word Extraction and Feature Extraction

For the next 4 models, discussed in sections 4.3.2 to 4.3.5, the following common steps and parameters are considered for pre-processing, word extraction, and feature extraction:

- Word extraction: Manually
- Feature extraction technique: MFDWC
- Frame-width: $N = 256$ samples
- Overlapping frames: $M = 100$ samples
- Hamming window: $\beta = 0.46$
- Number of filters: 20

The following subsection discusses the results obtained for various classification techniques.

4.3.2 Model 1: Dynamic Time Warping

Overall, this is our fourth model for speech recognition of isolated Gujarati words. For the first time, we are using a different feature extraction technique: mel-frequency discrete wavelet coefficients. The details of the dataset are as follows:

- Vocabulary: Digits 1-10 spoken in Gujarati
- Number of speakers: 4
- Sampling rate: 16,000
- Number of *.wav files: 40

Various parameters used for feature extraction, for this model, are as follows:

- Pre-emphasising: $\alpha = 0.95$
- Wavelet used: Coiflet (as shown in the Figure 4.9)
- Level of decomposition: 3
- Number of coefficient per frame: 10

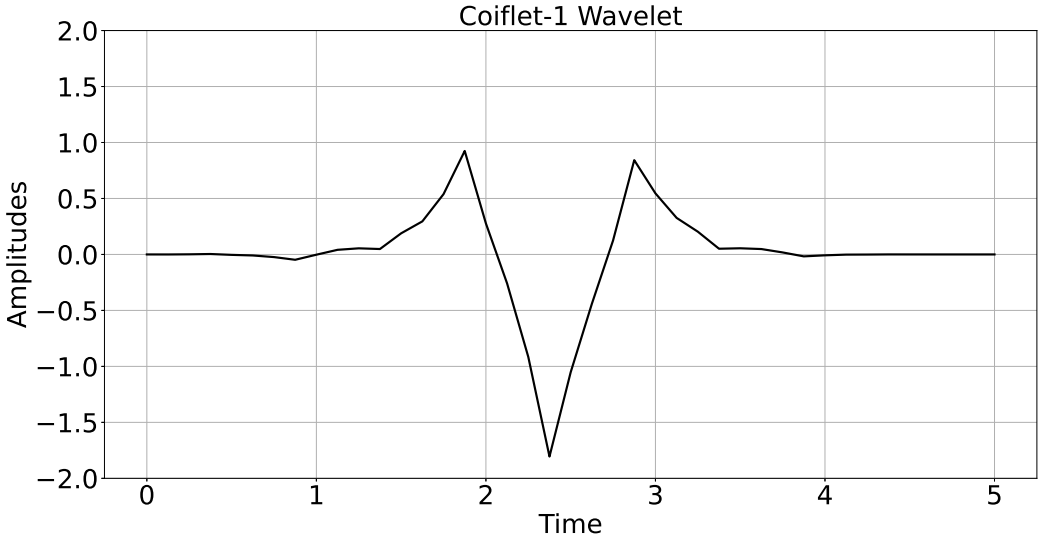

Figure 4.9: The plot of Coiflet-1 wavelet function

Speakers	**1**	**2**	**3**	**4**
1	10	8	6	8
2	8	10	7	7
3	6	7	10	10
4	8	7	10	10

Table 4.1: Number of words matching out of 10 using dynamic time warping

For the recognition, the dynamic time warping distance is determined between all the pairs of speakers, using the equation (4.1).

$$D(x_i, y_j) = |x_i - y_j| + \min\{D(x_i, y_{j-1}), D(x_{i-1}, y_{j-1}), D(x_{i-1}, y_j)\} \tag{4.1}$$

Here, i and j represent indices for the positions of feature vectors x and y respectively. The distance is measured between all the ten words spoken by one speaker and the ten words spoken by another speaker. Words spoken by two speakers are said to be matching if they give the minimum dynamic time warping distance. So for each pair of speakers, the number of words matching, out of 10, is determined.

The summary of these pairwise calculations is shown in the Table 4.1. Here, we observe that 8 digits of speaker 1 are matching with speakers 2 and 4, and 6 digits are matching with speaker 3. Similarly, for speaker 2, 7 digits are matching with those of speakers 3 and 4. Finally, all 10 digits of speaker 3 are matching with those of speaker 4. From this, we can conclude that 86% of the words are matching, as shown in our work [78]. This accuracy is slightly higher as compared to the results obtained in the section 3.6 as in [75]. This is due to the introduction of wavelets in the feature extraction technique.

Next, we will see one of the most important and successful works of our research, i.e., one based on multilayered perceptrons.

4.3.3 Model 2: Multilayered Perceptron

This is our fifth model for the speech recognition of isolated words spoken in the Gujarati language. The MFDWC features are classified using the multilayered perceptron trained with the more recent Adam algorithm [64] and with the ReLU activation function [63].

The details of the dataset are as follows:

- Vocabulary: Digits 1-10 spoken in Gujarati
- Number of speakers: 8
- Sampling rate: 16,000
- Number of *.wav files: 80

Various parameters used for feature extraction, for this model, are as follows:

- Pre-emphasising: $\alpha = 0.97$
- Wavelet used: Daubechies 1-6
- Level of decomposition: 2
- Number of coefficient per frame: 10

For feature extraction, we have used six different wavelets, one by one, at level 2. These six wavelets are Daubechies-1 to Daubechies-6. The output of the feature extraction step is a set of 80 feature vectors. The length of all the feature vectors is made equal to the median length of 603 using the cubic spline interpolation method.

For splitting the data in training and testing, out of a dataset of 80, 70 feature vectors are taken for training the multilayered perceptron, and the rest of the 10 feature vectors are taken for testing. This splitting is done using stratified random sampling, so that the test dataset contains all the digits from one to ten and at least one digit from each speaker.

For training, a multilayered perceptron, as shown in the Figure 4.10, with architecture $N^2_{603,78,10}$ is considered. It has 603 neurons in the input layer. This represents the median length of all the feature vectors. 10 neurons in the output layer representing the classes corresponding to 10 words using the one-hot encoding. Thus, the input layer has nodes

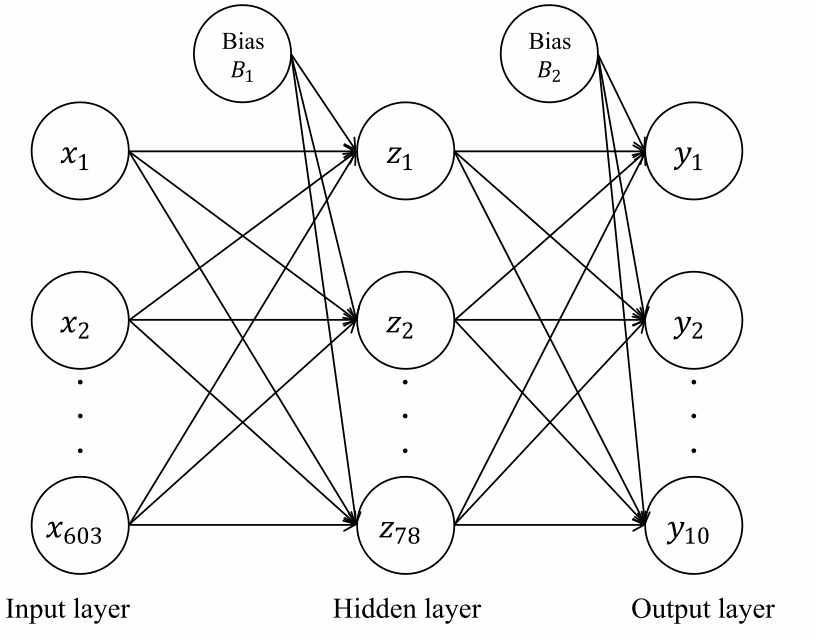

Figure 4.10: A multilayered network architecture used in Model 2

$\vec{x} = (x_1, x_2, \ldots, x_{603})$. The neurons in the hidden layers are $\vec{z} = (z_1, z_2, \ldots, z_{78})$. The output is $\vec{y} = (y_1, y_2, \ldots, y_{10})$.

To determine the number of neurons in the hidden layers, many researchers have proposed various techniques [79]. We have used the number of neurons in the hidden layer as the integer near to the square root of the product of the number of neurons in the input layer and the number of neurons in the output layer, as in [80].

The activation function used in the hidden layer is the rectified linear unit (ReLU) function given by equation (4.2).

$$f_h(x) = \max(0, x) \tag{4.2}$$

The activation function used in the output layer is the normalised exponential (softmax) function given by equation (4.3).

$$f_o(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{10} e^{x_j}}, 1 \leqslant i \leqslant 10 \tag{4.3}$$

The multilayered perceptron is trained with an error-back propagation algorithm. The weights are optimised using the Adam algorithm, using equation (4.4).

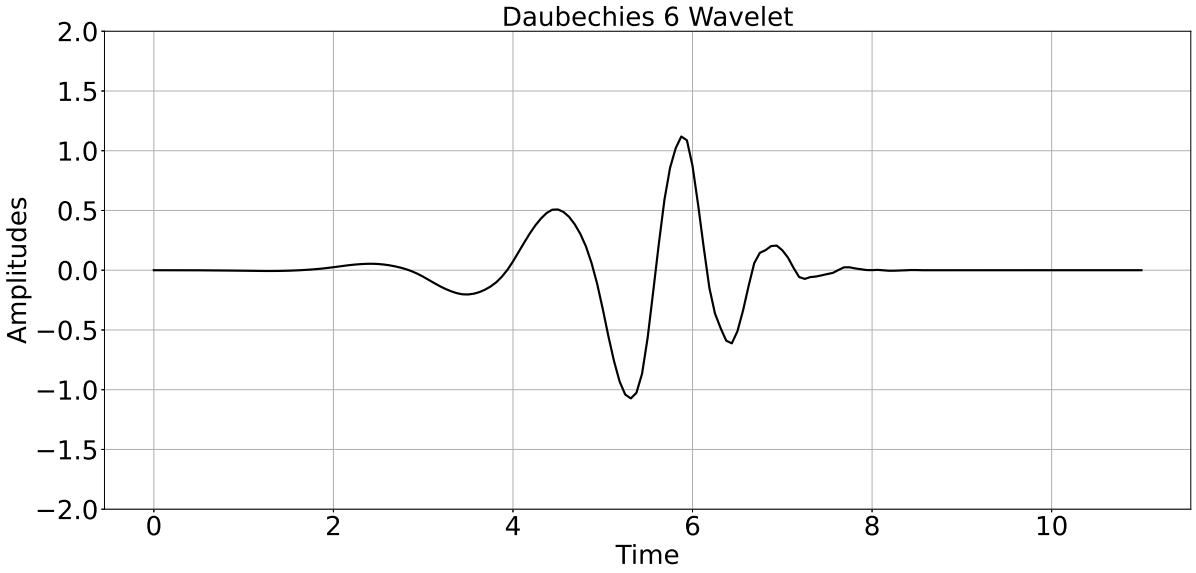

Figure 4.11: The plot of Daubechies-6 wavelet

$$\Delta w = -\frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \tag{4.4}$$

Here,

$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ is updated momentum at step t,
$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla w_t$ is momentum at step t,
$\beta_1 = 0.9$,
$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$ is used to keep history of gradients,
$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla w_t)^2$,
$\beta_2 = 0.999$,
$\eta = 0.001$ is a learning rate,
$\epsilon = 1 \times 10^{-8}$.

The cross-entropy loss (log-loss) function, given by equation (4.5), is used to find loss at the output layer of the multilayered perceptron.

$$L = -\frac{1}{70} \sum_{i=1}^{70} \sum_{k=1}^{10} y_{i,k} \log(p_{i,k}) \tag{4.5}$$

To achieve the best results, six different types of wavelets are used, six different multilayered perceptron neural networks are trained 20 times, and average accuracy is determined correspondingly. The results obtained for various wavelets are summarised in the Table 4.2. On average, 92% accuracy is achieved for the Daubechies-6 wavelet, which is the best among all the other wavelets; refer to [81].

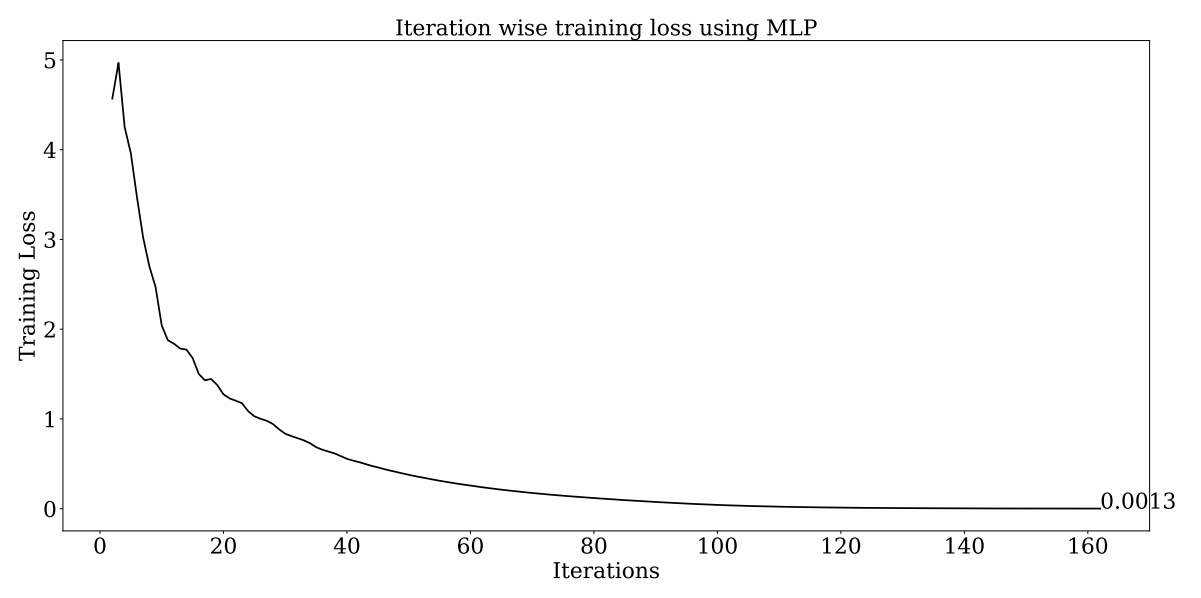

Figure 4.12: Decay in training loss with increase in iterations for Model 2

Name of wavelet	**Average accuracy (%)**	**Average train time (second)**	**Average number of iterations**
Daubechies - 1	47.50	2.95	100
Daubechies - 2	53.50	3.69	122
Daubechies - 3	58.00	3.60	121
Daubechies - 4	78.50	4.35	150
Daubechies - 5	85.00	4.74	158
Daubechies - 6	92.00	5.14	168

Table 4.2: Summary of results obtained for different wavelets (Model 2)

The plot of training progression for the Daubechies-6 wavelet for one such train is shown in the Figure 4.12. For the Daubechies-6 wavelet, as shown in Figure 4.11, on average, it took 5.14 seconds for the network to train with 168 iterations. The value of the log loss function at the end of the end of the final iteration is 0.0013. Then, these networks were tested with the ten test feature vectors. The accuracy is measured using a confusion matrix. One such confusion matrix for the Daubechies-6 wavelet is shown in the Figure 4.13.

Training with Leave-One-Out Cross Validation

For the model-2, we trained it using the leave-one-out cross validation. In this, the multilayered perceptron is trained with 79 feature vectors, leaving out 1 feature vector for testing. This is repeated 80 times, leaving one feature vector at a time. The cumulative confusion matrix is shown in the Figure 4.14. This gives a 92.13% test accuracy; refer to [81].

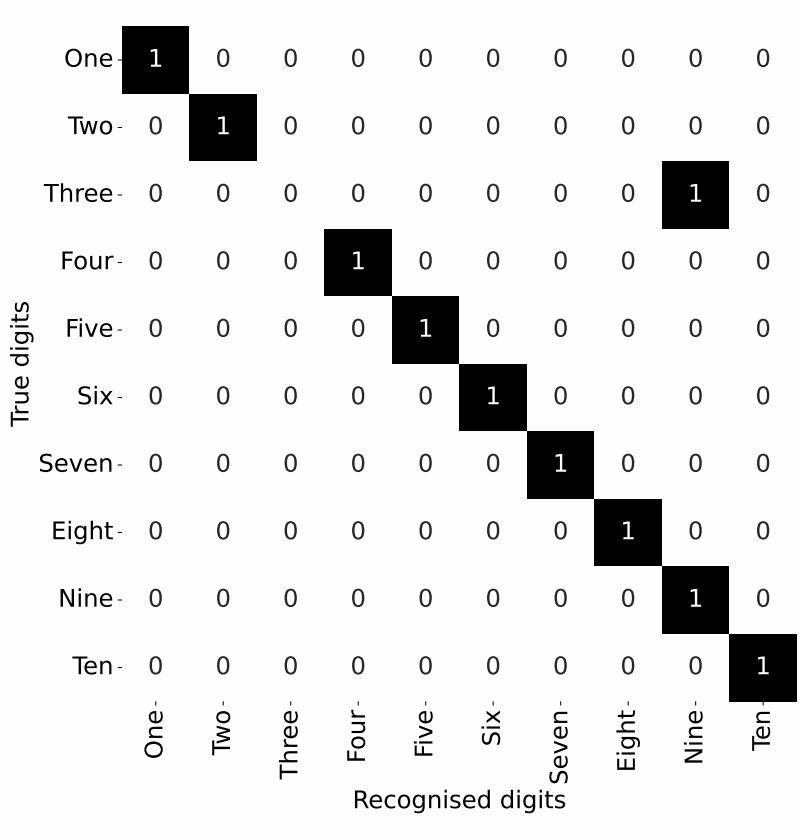

Figure 4.13: Confusion matrix of 10 test samples of feature vectors for Model 2

4.3.4 Model 3: Radial Basis Function Network

This is our sixth model for the speech recognition of isolated words spoken in the Gujarati language. The classification is done using the radial basis function network. This differs from previous implementations using a radial basis function network, in that the feature vectors are produced using MFCC. The details of the dataset are as follows:

- Vocabulary: Digits 1-10 spoken in Gujarati
- Number of speakers: 8
- Sampling rate: 16,000
- Number of *.wav files: 80

Various parameters used for feature extraction, for this model, are as follows:

- Pre-emphasising: $\alpha = 0.97$
- Wavelet used: Daubechies-6

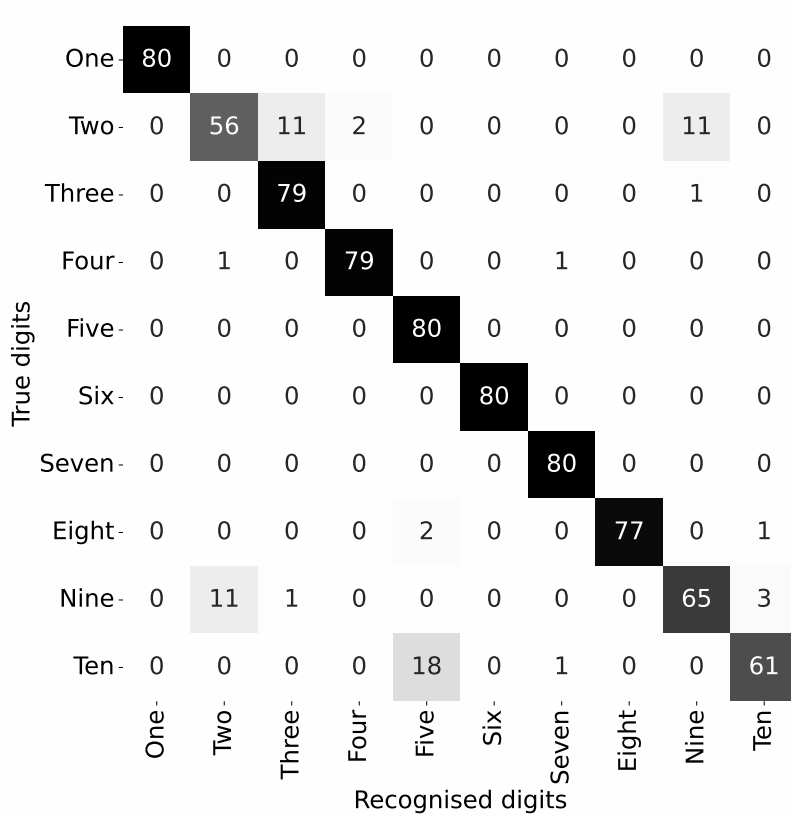

Figure 4.14: Confusion matrix with leave-one-out procedure for Model 2

- Level of decomposition: 2
- Number of coefficient per frame: 10

The features are extracted from all 80 speech files. The lengths of the feature vectors are made equal using the cubic spline interpolation method.

For training the radial basis function network, we first normalise the data using the equation (4.6).

$$z_i = \frac{x_i - \bar{x_i}}{s_i}, \tag{4.6}$$

Here, x_i is the i^{th} feature in the input data, $\bar{x_i}$ is the mean and s_i is the standard deviation of i^{th} feature.

After normalising the data, it is divided into training and testing by the stratified sampling method, with 70 feature vectors in training and 10 feature vectors in testing. After that, as shown in the Figure 4.15, a radial basis function network with architecture $N^2_{603,70,4}$ is constructed.

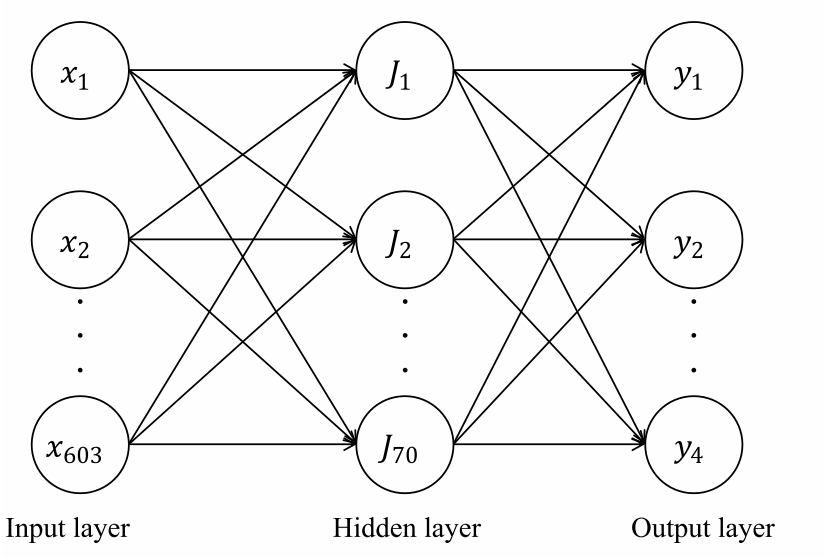

Figure 4.15: A radial basis function network architecture used in our work for Model 3

It has 603 neurons $(x_1, x_2, \ldots, x_{603})$ in the input layer, representing the median of the lengths of all the feature vectors, and 4 neurons (y_1, y_2, y_3, y_4) in the output layer, representing the digits one to ten in binary form. There are 70 neurons in the hidden layer $(J_1, J_2, \ldots, J_{70})$. The Gaussian activation function, given by equation (4.7), is applied to each neuron in the hidden layer.

$$\phi(r) = e^{\frac{-r^2}{2\sigma^2}} \tag{4.7}$$

Then, the centres and weights of the radial basis function network are selected randomly in the first iteration. Then, they are iteratively updated using equations (4.8) and (4.9).

$$w_{mi}^{k+1} = w_{mi}^{k} + \frac{2\eta_1}{70} \sum_{n=1}^{70} \sum_{i=1}^{4} e_{ni} \sum_{m=1}^{70} \phi\left(||\vec{x_n} - \vec{c_m}||\right) \tag{4.8}$$

$$\vec{c_m}^{\,k+1} = \vec{c_m}^{\,k} - \frac{2\eta_2}{70} \sum_{n=1}^{70} \sum_{i=1}^{4} e_{ni} \sum_{m=1}^{70} w_{mi} \phi'\left(\left\|\vec{x}_n - \vec{c_m}^{\,k}\right\|\right) \frac{\vec{x}_n - \vec{c_m}^{\,k}}{||\vec{x}_n - \vec{c_m}^{\,k}||} \tag{4.9}$$

To determine the optimum value of a spread parameter (Gaussian neuron bandwidth) σ, various radial basis function networks are trained by taking different values of σ. The value giving the minimum error for the loss function is considered. It is found that $\sigma = 23$ gave the least error of the loss function.

$$L = \frac{1}{70} \sum_{n=1}^{70} \sum_{i=1}^{4} \left[y_{ni} - \sum_{m=1}^{70} w_{mi} \phi\left(\|\vec{x}_n - \vec{c}_m\|\right) \right]^2 \tag{4.10}$$

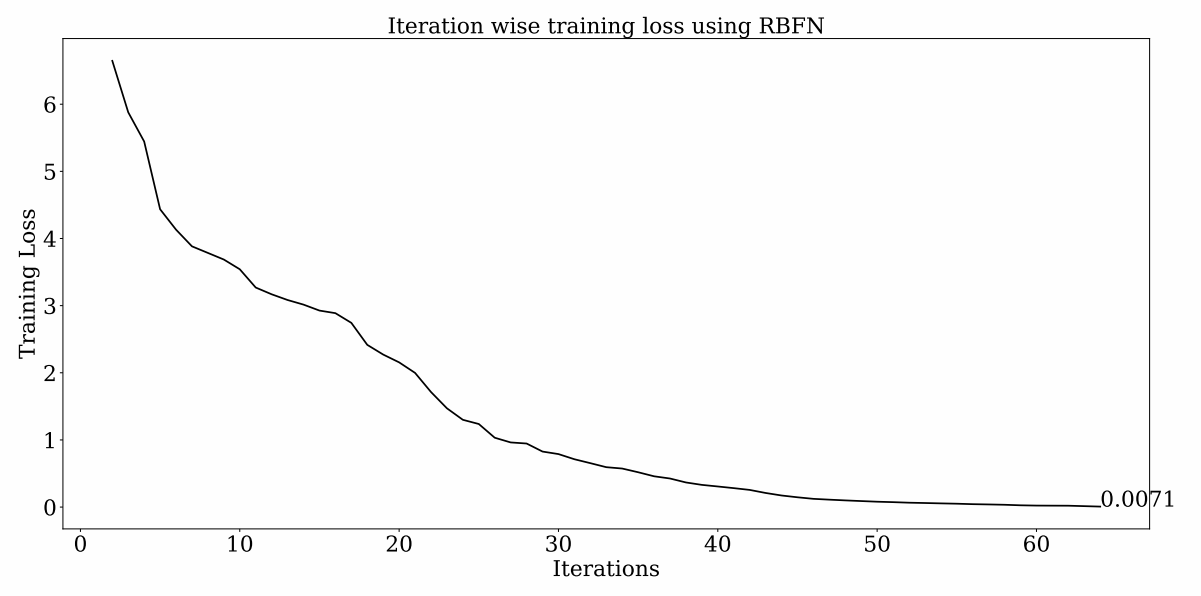

Figure 4.16: Decay in training loss with increase in iterations for Model 3

It takes average 0.54 seconds to achieve this accuracy. The progress of the training procedure is shown in the Figure 4.16. The training loss is determined by equation (4.10). At the final iteration, it is 0.0071.
Test accuracy of 90% is achieved for the Daubechies-6 wavelet, refer to [81].

Next, we will discuss the models and results obtained for speech recognition of isolated words by hidden Markov models (HMMs).

4.3.5 Model 4: Hidden Markov Models

This is the last model for speech recognition of isolated words in Gujarati language. In this model the hidden Markov models and Gaussian mixture models are used for the recognition part. The details of the dataset are as follows:

- Vocabulary: Digits 1-10 spoken in Gujarati
- Number of speakers: 8
- Sampling rate: 16000
- Number of *.wav files: 80

The initial steps of recording, pre-processing, feature extraction, and train-test splitting are similar to sections 4.3.3 and 4.3.4. The various parameters used for feature extraction in this model are as follows:

- Pre-emphasising: $\alpha = 0.97$
- Wavelet used: Daubechies 6
- Level of decomposition: 2
- Number of coefficient per frame: 13

After that, the left-to-right hidden Markov models are created for each word. The hidden states are the speech units hidden inside the recording, and the observations are the feature vectors. Different numbers of states from 2 to 7 are considered, and the accuracy is determined in each case.

The hidden Markov model parameters π_i and a_{ij} are initialised randomly. The parameter $b_j(o_k)$ is determined by using the probability distribution of feature vectors using Gaussian mixture models by equation (4.11).

$$b_j(o_k) = \sum_{m=1}^{M} c_{jm} \frac{1}{\sqrt{2\pi|\Sigma_{jm}|}} e^{-\frac{1}{2}(o_k-\mu_{jm})^T \Sigma_{jm}^{-1}(o_k-\mu_{jm})} \tag{4.11}$$

In this, the mean and variance of each feature vector for each dimension are determined. Out of 80 feature vectors, 70 feature vectors are used to train the parameters of the model. The parameters of hidden Markov models are trained and updated iteratively using the Baum-Welch algorithm. This is implemented using equations (4.12), (4.13), and (4.14).

$$\therefore \overline{\pi}_i = \gamma_1(i) \tag{4.12}$$

$$\therefore \overline{a}_{ij} = \sum_{t=1}^{T-1} \xi_t(i,j) \Bigg/ \sum_{t=1}^{T-1} \gamma_t(i) \tag{4.13}$$

$$\therefore \overline{b}_j(k) = \sum_{\substack{t=1 \\ O_t=v_k}}^{T} \gamma_t(j) \Bigg/ \sum_{t=1}^{T} \gamma_t(j) \tag{4.14}$$

The training time required to do this for different numbers of states is summarised in the second column of the Table 4.3. During testing, given an unknown word, these trained parameters are used to determine the model of the unknown word, and in this way, the spoken word is identified. The model is tested with 10 words, and the test accuracy obtained for different numbers of states is summarised in the third column of the Table 4.3. The confusion matrix is shown in Figure 4.17. The model is able to classify all the words correctly. Various performance measures like recall, precision, and F-measure are

	Original Dataset		Augmented Dataset	
Number of States in HMM	Training Time (sec)	Test Accuracy (%)	Training Time (sec)	Test Accuracy (%)
2	5.17	100	14.14	60
3	5.38	100	14.93	65
4	5.62	100	16.33	65
5	5.76	100	16.15	70
6	6.19	100	17.14	65
7	6.33	100	17.78	65

Table 4.3: Summary of results of models based on HMM

	Original dataset			Augmented dataset		
Digits	Recall	Precision	F-measure	Recall	Precision	F-measure
1	1	1	1	0.67	1	0.80
2	1	1	1	1	0.50	0.67
3	1	1	1	1	0.50	0.67
4	1	1	1	0.67	1	0.80
5	1	1	1	1	0.50	0.67
6	1	1	1	0.67	1	0.80
7	1	1	1	0.67	1	0.80
8	1	1	1	1	0.50	0.67
9	1	1	1	0.50	0.50	0.50
10	1	1	1	0.50	0.50	0.50

Table 4.4: Performance measures for Hidden Markov Models

summarised in columns 2-4 of the Table 4.4; refer to [82].

4.3.6 Models for Augmented Dataset

[A] Multilayered Perceptron

- **Preprocessing**:
  To check whether increasing the dataset will improve the recognition results, we have created the second dataset using the dataset of model-2 in section 4.3.3, an expanded version of the original dataset. It is expanded using data augmentation methods like amplifying the signal by some factor, de-amplifying the signal by some factor, adding noise to the signal, shrinking the signal in time, and stretching the signal in time. These variations are performed on all raw speech signals, and a dataset of 1200 total speech signals is created, containing 120 samples of each digit from one to ten. Then, preprocessing steps, viz., word extraction, feature

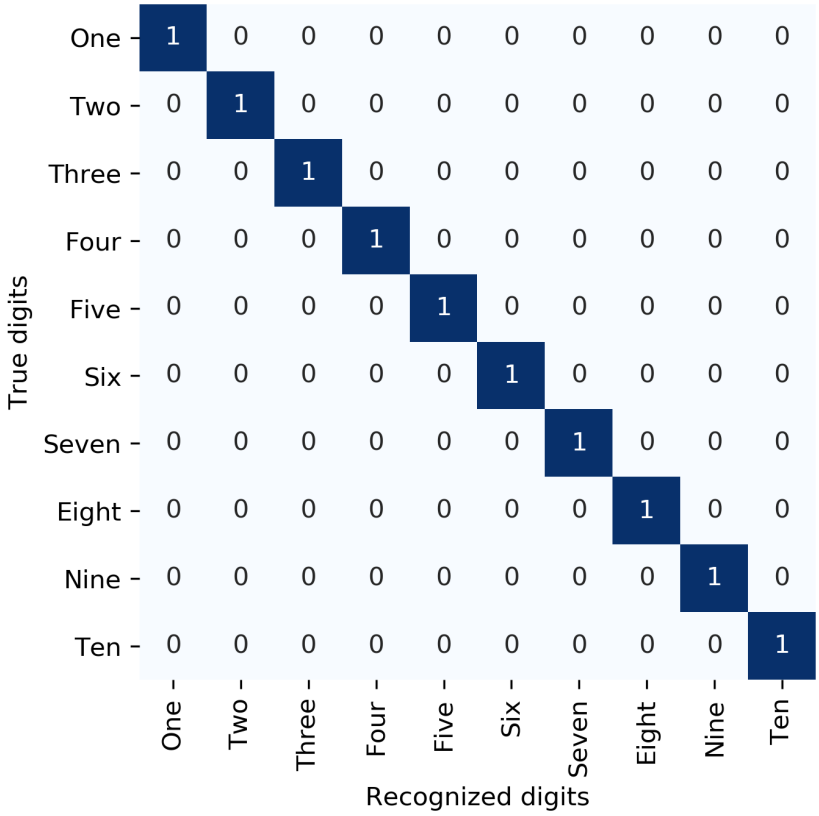

Figure 4.17: Confusion matrix for the classification using HMM

extraction, and making the length of all the feature vectors equal, similar to section 4.3.3, are performed.

- **Training and classification**:
  For a multilayered perceptron, the dataset is divided into a train set and a test set. The train set consists of 1000 feature vectors, and the remaining 200 feature vectors are part of the test set. The Daubechies-6 wavelet is used for wavelet decomposition. Then, the multilayered perceptron with network architecture $N^4_{603,80,80,40,10}$, consisting of three hidden layers of 80, 80, and 40 neurons, respectively, is trained using the Adam algorithm. The ReLU activation function is used in hidden layers, and the normalised exponential activation function is used in the output layer.

  During training, it took 1625 iterations and 161 seconds to achieve a 0.04 cross-entropy loss value. This model is tested with the remaining 200 feature vectors. The confusion matrix for the classification results, as shown in the Figure 4.18, suggests 85% testing accuracy. Moreover, various performance measures like precision, F-score, and recall are calculated for each class. They are summarised in the Table 4.5; refer to [81].

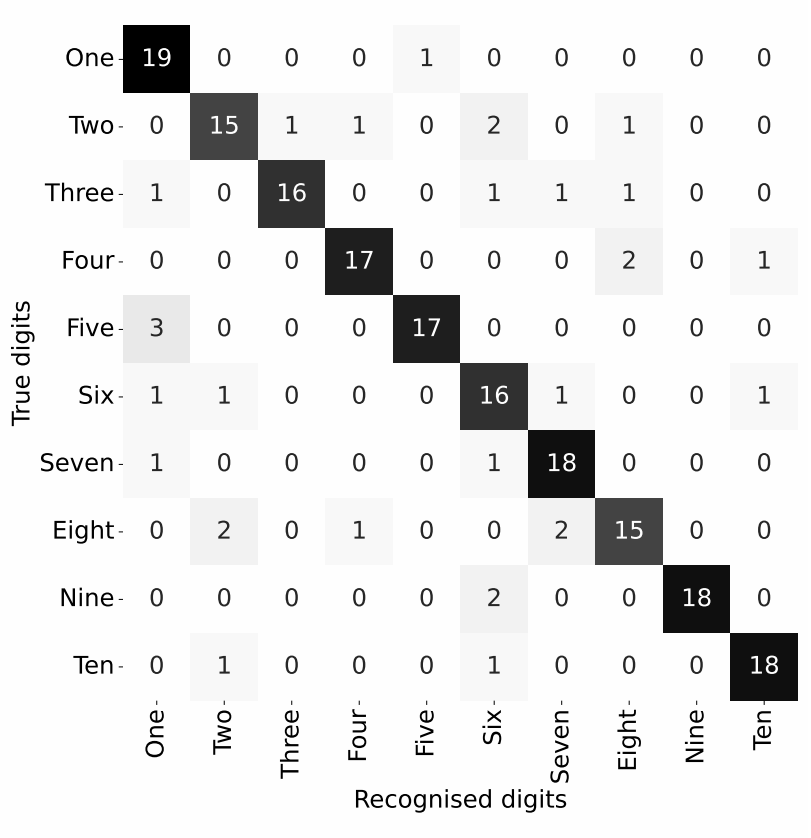

Figure 4.18: Confusion matrix for 200 test patterns for the augmented dataset

Recognised Digit	**Recall**	**Precision**	**F-Score**
One	0.76	0.95	0.84
Two	0.79	0.75	0.77
Three	0.94	0.80	0.86
Four	0.89	0.85	0.87
Five	0.94	0.85	0.89
Six	0.70	0.80	0.74
Seven	0.82	0.90	0.86
Eight	0.79	0.75	0.77
Nine	1.00	0.90	0.95
Ten	0.90	0.90	0.90

Table 4.5: Accuracy measures for multilayered perceptron for the augmented dataset

Classification algorithm	Test Accuracy
Dynamic time warping	86.00%
Multilayered perceptron	92.00%
Radial basis function network	90.00%
Hidden Markov model	100.00%

Table 4.6: Accuracy obtained for Speech recognition of isolated words using MFDWC feature extraction.

[B] Hidden Markov Model

- **Preprocessing:**

  The model discussed in the section 4.3.5 is also used for the augmented data. In this case, the dataset of section 4.3.5 is expanded using the data augmentation technique of amplifying and de-amplifying. Moreover, noise is also added to the augmented data set. The augmented dataset consists of 160 speech files. Then features are extracted from each of these files, resulting in 160 feature vectors.

- **Training and classification:**

  Out of 160, 140 feature vectors are used for training to determine the hidden Markov model parameters. The time required to train for different numbers of states is summarised in the fourth column of the Table 4.3. The model is tested with 20 feature vectors, and the test accuracy obtained for different numbers of states is summarised in the fifth column of the Table 4.3. The confusion matrix is shown in the Figure 4.19. The model is able to classify 70% words correctly for 5 states. Various performance measures like recall, precision, and F-measure are summarised in columns 5-7 of the Table 4.4; refer to [82].

4.3.7 Accuracy Comparison for Isolated Words based on MFDWC

In this section, we discussed four models of speech recognition of isolated Gujarati digits using four different classification techniques. The accuracies obtained for these approaches are summarised in the Table 4.6. We conclude that hidden Markov models and multilayered perceptrons yield the highest accuracy for the recognition of isolated words using MFDWC.

Comparing these results with the results of chapter 3, we conclude that overall feature extraction using MFDWC yields more recognition accuracy as compared to MFCC. The proposed use of wavelets, together with machine learning techniques, is successful. This

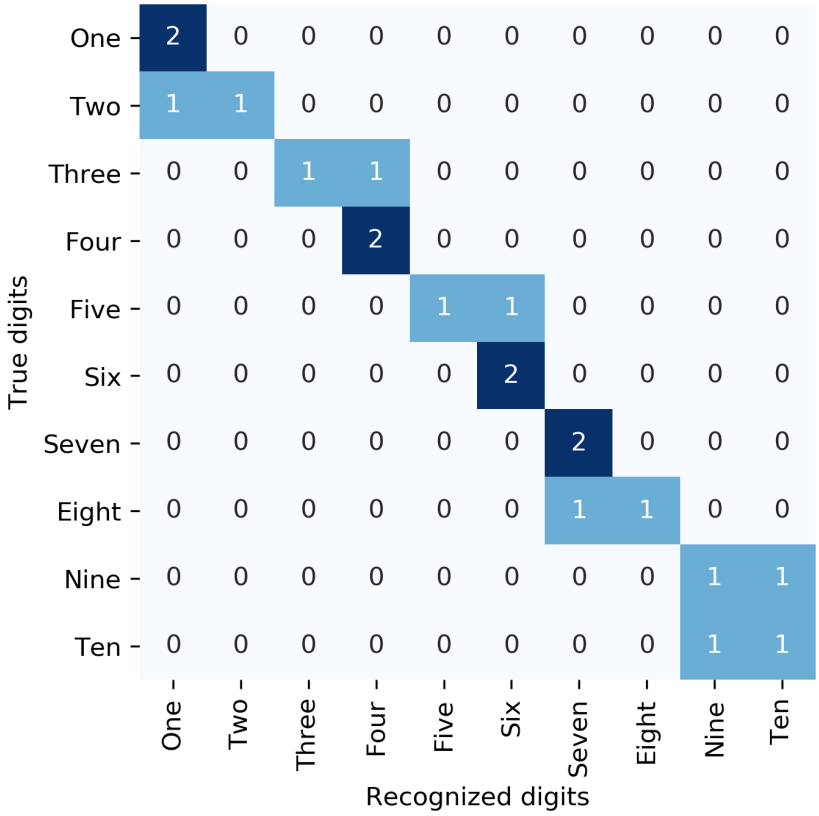

Figure 4.19: Confusion matrix using HMM for augmented dataset

Classification algorithm	Accuracy
Multilayered perceptron	85%
Hidden Markov model	70%

Table 4.7: Accuracy obtained for Speech recognition of isolated words for the augmented data.

is because wavelets are better for analysing non-stationary signals like speech.

Since a large dataset may give better results instead of overfitting, we included the step of considering the expanded dataset. The dataset is expanded using various augmentation techniques. We even considered adding additional random noise to the dataset. For this large dataset, we have chosen the two best algorithms from the Table 4.6: multilayered perceptrons and hidden Markov models. Both classification techniques are trained and tested on the augmented dataset. The results are summarised in the Table 4.7.

We obtained fairly nice results with this approach in the case of multilayered perceptron, but not better as compared to the original dataset. We conclude that the multilayered perceptron yields better accuracy, as compared to the hidden Markov model, for the augmented dataset. This means that it is able to handle the variations in the speech data

nicely as compared to the hidden Markov model. This is due to the fact that we can adjust the model parameters, like hidden layers and hidden neurons. This gives better generalisation, so that the model is able to handle the variations in the large dataset. Hence, for real-world situations, the approach of multilayered perceptron is recommended.

In the next section of this chapter, we will discuss two more models performed by us for the speech recognition of Gujarati. These two models are different from all the models discussed till now. They are models based on the recognition of continuous sentences.

4.4 Models for Recognition of Continuous Sentences

Till now, we have seen seven models for speech recognition in the Gujarati language. All seven models are based on speech recognition for isolated words. In these models, the recordings are isolated words. The words are extracted from the recording manually by identifying the unvoiced regions in the speech. Now, we will discuss speech recognition for continuous sentences. Here, recording is done in a continuous manner. The words are extracted from the sentences automatically by short-term autocorrelation method.

4.4.1 Word Extraction using Short-Term Autocorrelation

In the digital recording of speech, the smallest continuous unit is the word. Because, as we speak, there is a tiny gap between two words, but there is no gap between two letters of the word. Hence, it is essential to process the digital speech signal word-wise. The recording can be either word-wise or sentence-wise. So, in the later case, if the recording is sentence-wise, it is important to break a sentence into words. There are many useful algorithms and techniques for this task. We have used a short-term autocorrelation technique (STAC) discussed in the section 2.3.

Let us consider an example of a real-world speech signal shown in the Figure 4.20. This speech signal consists of a total 17 words. We can observe these 17 words easily due to their amplitudes. Using the short-term autocorrelation method as explained in the section 2.3, we can determine the unvoiced zone in this signal, as shown in the Figure 4.21, with lag $n = 1$ and $N = 400$ samples per frame. This corresponds to a 25-millisecond window. The short-term auto-correlations can be any positive number, which is normalised in interval [0, 1], as shown in the Figure 4.21. So, Figure 4.21 clearly depicts the distinction between 17 spoken words and unvoiced speech.

From this, we can determine whether the particular frame is the voiced frame or the

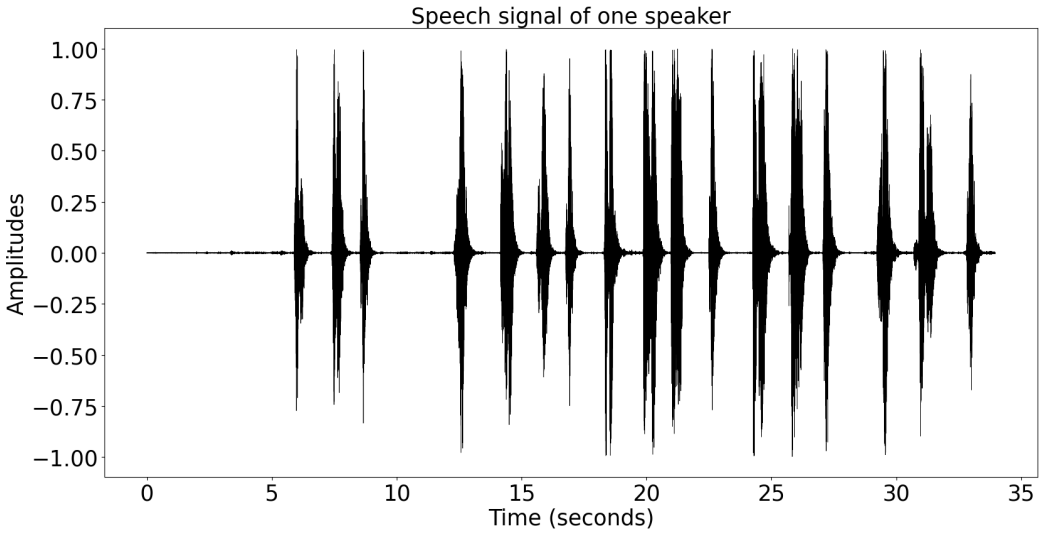

Figure 4.20: A plot of a speech signal having 17 words

unvoiced frame. Frames with an autocorrelation near to zero, are classified as silenced frames. This way, the silenced zone between two words in the sentence can be identified. Using this, we can break a sentence in words. These words are further used for feature extraction.

The words, once extracted, are then processed further to get feature vectors using mel-frequency discrete wavelet coefficients. Then, their lengths are made equal to the median lengths, using the cubic spline interpolation method. After that, they are used for training and testing using two different models: multilayered perceptrons and the hidden Markov model. The flowchart of the process of ASR for continuous sentences is as shown in the Figure 4.22. In the subsequent subsections, we will see details of each model and their performance results.

4.4.2 Model 1: Multilayered Perceptron

For this model, we have considered a small vocabulary consisting of 13 words, different from digits 1-10. Using these 13 words, 5 sentences are formed, which are shown in the Figure 4.23. These sentences, spoken by six speakers, are recorded. The details of the dataset are as follows:

- Vocabulary: 13 words
- Recording: 5 sentences consisting of 17 words, along with repeated ones (refer Figure 4.23)

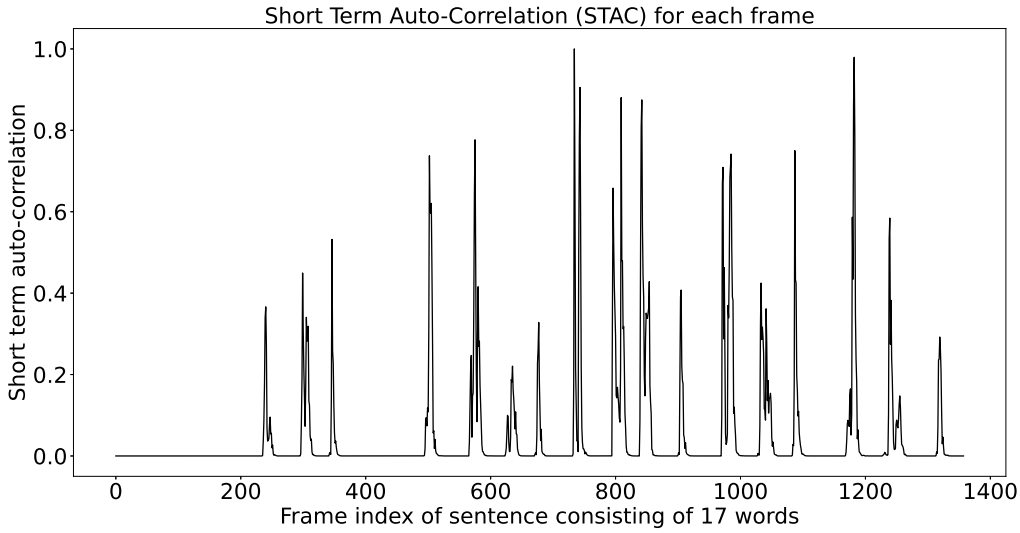

Figure 4.21: Short-term auto-correlation (normalised) of the signal shown in the Figure 4.20. The spikes shows the voiced part corresponding to 17 sentence.

Figure 4.22: Flowchart of process of ASR for continuous sentences

- Number of speakers: 6
- Sampling rate: 16,000
- Number of *.wav files: 6

To extract individual words from the recorded sentences, we have used the short-term autocorrelation method. This is performed using the equation (4.15).

$$a_n = \sum_{k=0}^{N-1} x_k x_{k-n} \tag{4.15}$$

We applied short-term autocorrelation with lag $n = 1$. This way, the unvoiced region is identified, and using the proposed algorithm, the 17 words are extracted. One of these words is shown in the Figure 4.24.

After extracting individual words from the sentences of the speech, the feature vectors are determined using the following parameters:

- Feature extraction technique: MFDWC

નદી વહે છે.
સૂરજ પૂર્વમાં ઊગે છે.
બતક પાણીમાં તરે છે.
કમળ ખીલે છે.
નિરજ જાગે છે.

Figure 4.23: 5 sentences having 17 words used for the models of continuous speech recognition using multilayered perceptrons

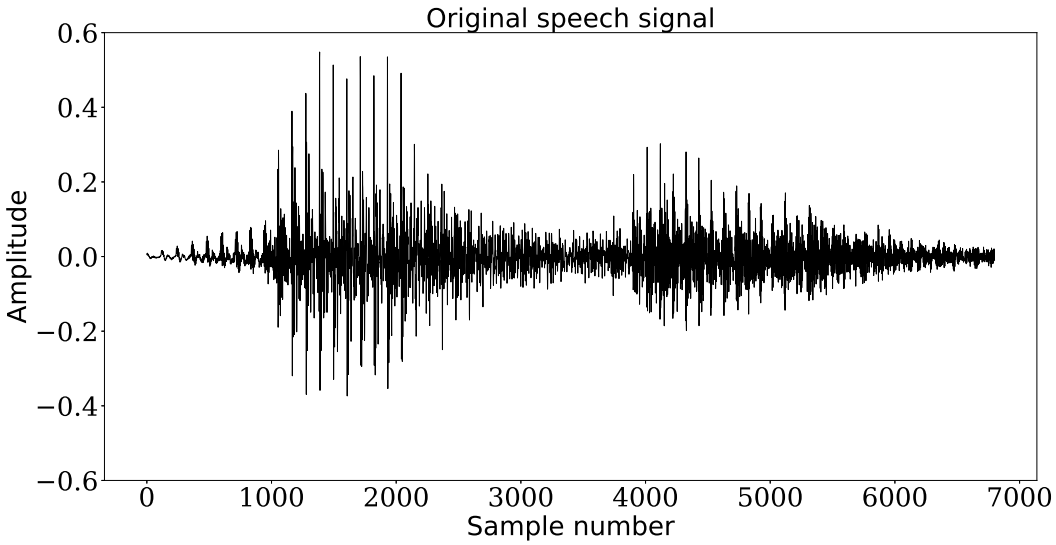

Figure 4.24: Plot of recording of word extracted using short-term autocorrelation

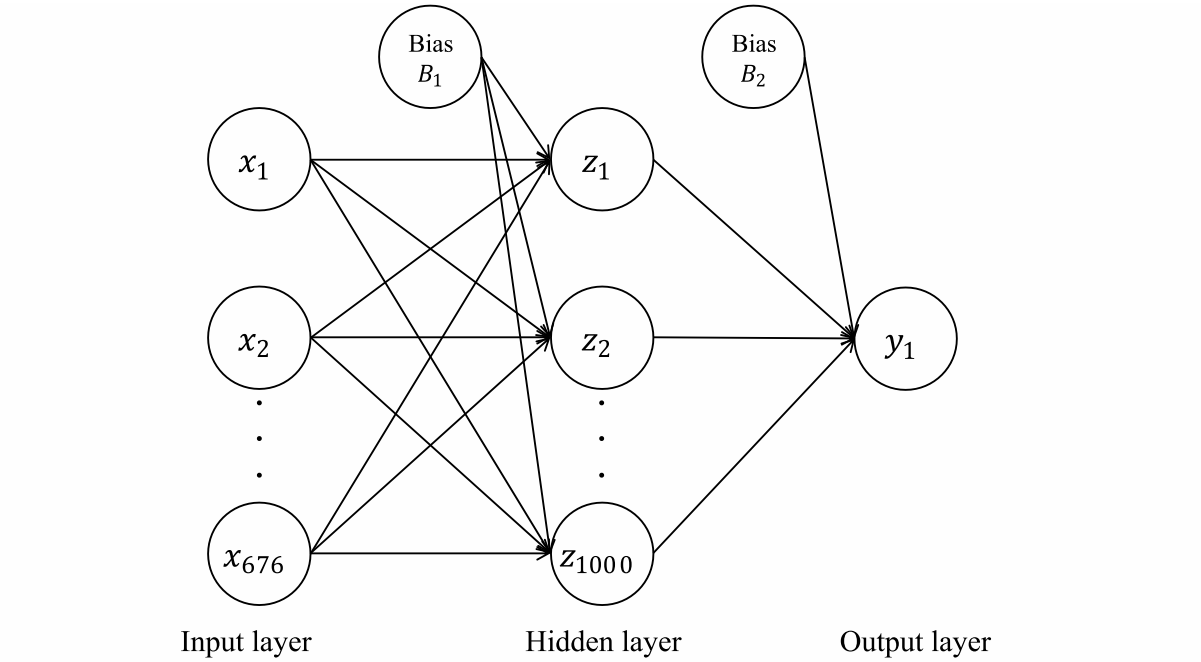

Figure 4.25: Multilayered perceptron architecture for the speech recognition of continuous words

- Pre-emphasising: $\alpha = 0.95$
- Frame-width: $N = 256$ samples
- Overlapping frames: $M = 100$ samples
- Hamming window: $\beta = 0.46$
- Number of filters: 20
- Wavelet used: Daubechies 6
- Level of decomposition: 2
- Number of coefficient per frame: 13

The lengths of all the feature vectors are made equal to the median length with the help of the cubic spline interpolation method. Then, the feature extraction is done using the MFDWC. In feature extraction, the Daubechies-6 wavelet is used at level 2.

Further, a multilayered perceptron with architecture $N^2_{676,1000,1}$ is considered, as shown in the Figure 4.25. 676 neurons in the input layer $\vec{x} = (x_1, x_2, \ldots, x_{676})$ represents the number of features in the input vectors, and one neuron, y, in the output layer represents the word label. The hidden layer consists of 1000 neurons $\vec{z} = (z_1, z_2, \ldots, z_{1000})$. The

activation function used in the output layer is the normalised exponential (softmax) function given by equation (4.16).

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{13} e^{x_j}}, 1 \leqslant i \leqslant 13 \tag{4.16}$$

For training the multilayered perceptron, feature vectors from all 17 words are used. For testing, separate sentences are created from the same vocabulary.

The multilayered perceptron is trained with two different approaches: one using the gradient descent method and another using the Adam algorithm.

1. **Training with Gradient Descent method:** For this approach of the gradient descent method, the logistic activation function is used in the hidden layer. It is given by equation (4.17).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4.17}$$

In this case, the change in weights of the multilayered perceptron are determined using equations (4.18) and (4.19).

$$v_j^{(k+1)} = v_j^{(k)} + \eta_o \left(d - y\right) f_o' \left(\sum_{j=0}^{1000} v_j^{(k)} z_j\right) z_j \tag{4.18}$$

$$w_{ji}^{(k+1)} = w_{ji}^{(k)} + \eta_h \left(d - y\right) f_o' \left(\sum_{j=0}^{1000} v_j z_j\right) \sum_{j=0}^{1000} v_j f_h' \left(\sum_{i=0}^{676} w_{ji}^{(k)} x_i\right) x_i \tag{4.19}$$

2. **Training with Adam algorithm:** For the second approach of the Adam algorithm, the rectified linear unit activation function is used in the hidden layer. It is given by equation (4.20).

$$f(x) = \max(0, x) \tag{4.20}$$

In this case, the weights of the multilayered perceptron are determined using equation (4.21).

$$\Delta w = -\frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{4.21}$$

Here,

$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ is updated momentum at step t,

$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\, \nabla w_t$ is momentum at step t,

$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$ is used to keep history of gradients,

$v_t = \beta_2 v_{t-1} + (1 - \beta_2)\, (\nabla w_t)^2$,

$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\eta = 0.001$, $\epsilon = 1 \times 10^{-8}$.

The training loss, for both approaches, is determined using cross-entropy (log-loss) loss function, given by equation (4.22).

$$L = -\frac{1}{102} \sum_{i=1}^{102} \sum_{k=1}^{13} y_{i,k} \log(p_{i,k}) \tag{4.22}$$

The results obtained for both approaches are summarised in Table 4.8. The trained networks are tested with unknown speech consisting of sentences different from the sentences trained but having the same vocabulary. After successfully recognising the speech sentence, our model gives text output in Gujarati, refer to [83]. It can be observed that gradient descent-based training with a logistic activation function is giving better results.

Algorithm	Activation function	Number of Iterations	Training time (sec)	Training loss	Testing accuracy
Gradient descent	Sigmoid	956	134	0.00742	84.62 %
Adam	ReLU	1098	29	0.00042	76.92 %

Table 4.8: Summary of models of continuos speech recognition using ANN

4.4.3 Model 2: Hidden Markov Models

In the last chapter, we mentioned the model using HMM for isolated words. We used a similar approach for continuous sentences. In this model, five long sentences with a total of 32 words are selected to generate the data. Out of 32 words, 16 are unique. These 5 sentences are shown in the Figure 4.26. The details of the dataset are as follows:

- Vocabulary: 16 words
- Recording: 5 sentences consisting of 32 words with all repetitions (refer Figure 4.26)
- Number of speakers: 10
- Sampling rate: 8000
- Number of *.wav files: 10

ભારત દેશ માં ગુજરાત રાજ્ય આવેલ છે.
ગુજરાત રાજ્ય માં ગુજરાતી ભાષા બોલાય છે.
ગુજરાત રાજ્ય નું પાટનગર ગાંધીનગર છે.
ગુજરાત રાજ્ય નું પાટનગર દિલ્લી નથી.
વડોદરા ગુજરાત રાજ્ય માં આવેલ છે.

Figure 4.26: 5 sentences having 32 words used for the models of continuous speech recognition using hidden Markov models

The words are extracted from sentences automatically using STAC. This produces 32 separate vector sequences for each file. Overall, there are 320 different vector sequences representing words extracted from the sentences.

In the next step, the features are extracted from words using the MFDWC. For this, the following parameters are considered:

- Feature extraction technique: MFDWC
- Pre-emphasising: $\alpha = 0.97$
- Frame-width: $N = 256$ samples
- Overlapping frames: $M = 100$ samples
- Hamming window: $\beta = 0.46$
- Number of filters: 20
- Wavelet used: Daubechies-6
- Level of decomposition: 2
- Number of coefficient per frame: 13

The lengths of these feature vectors are made equal to the median length $M = 8000$, using the cubic spline interpolation method. Further, a 5-state hidden Markov model is built for each unique word. The parameters of the hidden Markov model π_i and a_{ij} are initialised randomly. The observation probabilities $b_j(o_k)$ are determined using the Gaussian mixture model by equation (4.23).

$$b_j(o_k) = \sum_{m=1}^{M} c_{jm} \frac{1}{\sqrt{2\pi|\Sigma_{jm}|}} e^{-\frac{1}{2}(o_k-\mu_{jm})^T \Sigma_{jm}^{-1}(o_k-\mu_{jm})} \tag{4.23}$$

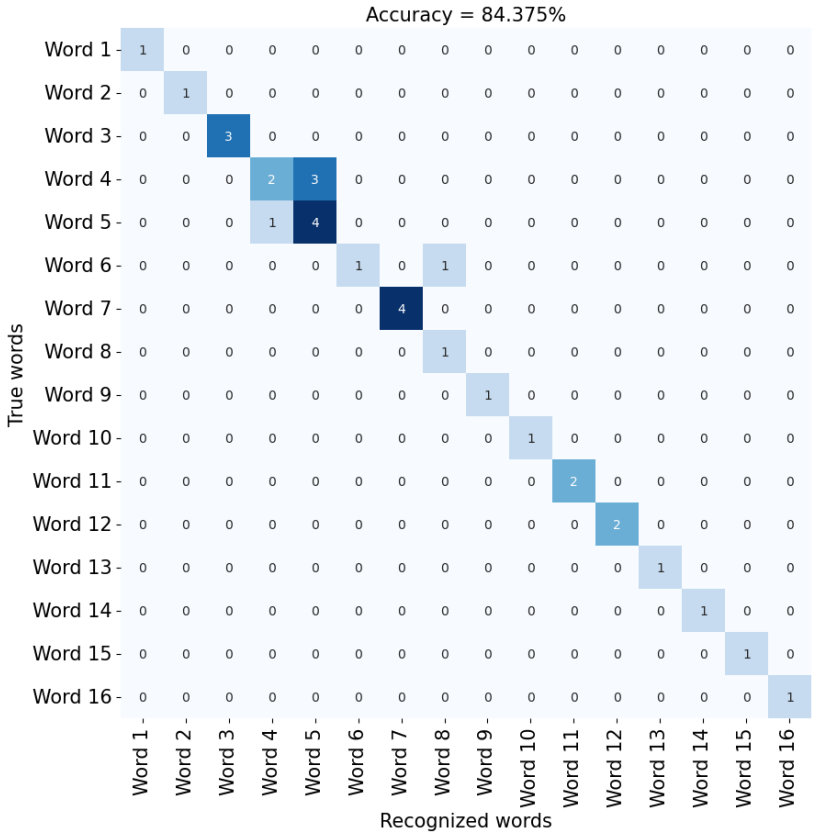

Figure 4.27: Confusion matrix for 16 words in the model of continuous speech recognition using HMM

The dataset is divided into training and testing using the stratified sampling method. Out of 320 feature vectors, 288 are used to train the model using the Baum-Welch algorithm, using equations (4.24), (4.25), and (4.26). The network is trained in 19.47 seconds.

$$\therefore \overline{\pi}_i = \gamma_1(i) \tag{4.24}$$

$$\therefore \overline{a}_{ij} = \sum_{t=1}^{T-1} \xi_t(i,j) \Bigg/ \sum_{t=1}^{T-1} \gamma_t(i) \tag{4.25}$$

$$\therefore \overline{b}_j(k) = \sum_{\substack{t=1 \\ O_t=v_k}}^{T} \gamma_t(j) \Bigg/ \sum_{t=1}^{T} \gamma_t(j) \tag{4.26}$$

The remaining 32 feature vectors are used for testing. Overall, 84.38% accuracy is obtained for the text dataset. The confusion matrix is as shown in the Figure 4.27. Various accuracy measures are shown in the Table 4.9.

Words	Test Patterns	Precision	Recall	f1-score
Word 1	1	1.00	1.00	1.00
Word 2	1	1.00	1.00	1.00
Word 3	3	1.00	1.00	1.00
Word 4	5	0.67	0.40	0.50
Word 5	5	0.57	0.80	0.67
Word 6	2	1.00	0.50	0.67
Word 7	4	1.00	1.00	1.00
Word 8	1	0.50	1.00	0.67
Word 9	1	1.00	1.00	1.00
Word 10	1	1.00	1.00	1.00
Word 11	2	1.00	1.00	1.00
Word 12	2	1.00	1.00	1.00
Word 13	1	1.00	1.00	1.00
Word 14	1	1.00	1.00	1.00
Word 15	1	1.00	1.00	1.00
Word 16	1	1.00	1.00	1.00
	32			

Table 4.9: Precision, recall and f1-score for 16 words in the model of continuous speech recognition using HMM

4.4.4 Accuracy Comparison for Continuous Sentences based on MFDWC

For the speech recognition of continuous Gujarati sentences, we have proposed two models, in which the words are extracted automatically from the recorded sentences using short-term autocorrelation. For the classification of features of extracted words, we have used two algorithms of multilayered perceptron: gradient descent and the Adam algorithm. We also tried with hidden Markov models. The accuracies obtained are summarised in the Table 4.10. We conclude that the gradient descent method performs better than Adam and slightly better than the hidden Markov model.

Classification algorithm	**Accuracy**
Multilayered perceptron (Gradient descent)	84.62%
Multilayered perceptron (Adam)	76.92%
Hidden Markov model	84.38%

Table 4.10: Accuracy obtained for speech recognition of continuous sentences.

4.5 Conclusion

This chapter consists of six overall models for automatic speech recognition in Gujarati. The features from the speech are extracted using MFDWC. These models are divided into two parts: recognition of isolated words and recognition of continuous sentences.

For the recognition of isolated words, we have used four models based on dynamic time warping, multilayered perceptron, radial basis function network, and hidden Markov model. It is observed that the multilayered perceptrons and hidden Markov models performed better for the recognition of isolated words. These two best models are further used on the expanded dataset using data augmentation techniques. In this, due to better generalisation, multilayered perceptron performed better.

For the recognition of continuous sentences, words are extracted from the speech automatically using short-term autocorrelation. We have used two kinds of models for recognition. One is a multilayered perceptron with the gradient descent method and Adam algorithm, and the other is a probabilistic modelling approach using the hidden Markov model. Multilayered perceptrons trained with the gradient descent method performed better.

Overall, multilayered perceptron and hidden Markov models are best for speech recognition in Gujarati. In the next chapter, we will use these two models in ensemble learning to see if their accuracy will increase or not.