Chapter 4

Speaker Segmentation Using Autoassociative Neural Networks

In this chapter, a new approach for speaker segmentation is proposed using autoassociative neural network. This method extracts the speaker specific information from the mel frequency cepstral coefficients (MFCC). The distribution capturing ability of the autoassociative neural network (AANN) model is exploited to detect the speaker change points. AANN model for capturing the distribution of acoustic feature vectors is given in Section 4.2. The MFCC feature extraction is described in Section 4.3. The proposed algorithm for speaker change detection is presented in Section 4.4. Section 4.5 presents the experimental results and the comparison of the proposed method with the existing methods.

4.1 Introduction

The significant algorithms used in the segmentation of audio data records are the metric based segmentation approaches, in which two adjacent windows are selected from the speech stream, and their dissimilarities are assessed by a distance function of their contents. Then the system locates a changing mark in the point in which the dissimilarity is high. Depending on the application, the analysis window may overlap or not. Metric based methods do not require any prior knowledge on the number of speakers, their identities, or signal characteristics. In this chapter, a new approach for speaker
segmentation is proposed using AANN. This method extracts the speaker specific information from MFCC. The speaker change points are detected using the distribution capturing ability of the AANN model. In this work, we have considered the problem of segmenting the speech of two speakers. The problem consists of automatically marking the periods of time in which every speaker is talking. This work formulates a new speaker change detection algorithm, which can detect speaker changes with speech segments of short duration. Moreover, this algorithm works without any prior knowledge of the identity of speakers, and hence, it is unsupervised.

4.2 AANN Model for Capturing the Distribution of Feature Vectors

Autoassociative neural network models are feed forward neural networks performing an identity mapping of the input space, and are used to capture the distribution of the input data [25], [138]. Limitation of principal component analysis (PCA) to represent an input space using a linear subspace motivated the researchers to investigate a method of projecting the input data onto a non-linear subspace using AANN models [139], [140]. A three layer AANN consists of three layers namely input layer, hidden layer and output layer. An AANN is a feed forward network with the desired output being the same as the input vector. Therefore, the number of units in the input and output layers are equal. The number of hidden layers and the number of units in each hidden layer depend on the problem.

A three layer AANN model clusters the input data in the linear subspace, whereas a five layer AANN model captures the non-linear subspace passing through the distribution of the input data. Studies on three layer AANN models show that the non-linear activation function at the hidden units clusters the input data in a linear subspace
Theoretically, it was shown that the weights of the network will produce small errors only for a set of points around the training data [141]. When the constraints of the network are relaxed in terms of layers, the network is able to cluster the input data in the non-linear subspace. Hence, a five layer autoassociative neural network model as shown in Fig. 4.1 is used to capture the distribution of the feature vectors in our study.

![Fig. 4.1: A five layer AANN model.](image)

Let us consider the five layer AANN model shown in Fig. 4.1, which has three hidden layers. The processing units in the first and third hidden layers are non-linear, and the units in the second compression/hidden layer can be linear or non-linear. As the error between the actual and the desired output vectors is minimized, the cluster of points in the input space determines the shape of the hyper surface obtained by the projection onto the lower dimensional space. The second and fourth layers of the network have more units than the input layer. The third layer has fewer units than the first or fifth. The activation functions at the second, third and fourth layers are non-linear. The structure of the AANN model used in our study is $19L_u \ 38N_u \ 5N_u \ 38N_u \ 19L_u$, where $L_u$ denotes a linear unit and $N_u$ denotes a non-linear unit. The non-linear output function for each unit is $\tanh(s)$, where $s$ is the activation value of
The standard back propagation learning algorithm [142] is used to adjust the weights of the network to minimize the mean square error for each feature vector. The AANN captures the distribution of the input data depending on the constraints imposed by the structure of the network, just as the number of mixtures and Gaussian functions do in the case of Gaussian mixture model. The choice of parameters such as feature vectors, initial weights and structure of AANN is not very critical, as variation of these parameters does not affect the performance of the system abruptly [143].

![Fig. 4.2: Distribution capturing ability of AANN model. (a) Artificial 2 dimensional data. (b) 2 dimensional output of AANN model. (c) Probability surfaces realized by the network.](image)

In order to visualize the distribution better, one can plot the error for each input data point in the form of some probability surface as shown in Fig. 4.2. The error $e_i$ for the data point $i$ in the input space is plotted as $p_i = \exp(-e_i/\alpha)$, where $\alpha$ is a constant. Note that $p_i$ is not strictly a probability density function, but we call the
resulting surface as probability surface. The plot of the probability surface shows a large amplitude for smaller error $e_i$, indicating better match of the network for that data point. The constraints imposed by the network can be seen by the shape of the error surface in both the cases. One can use the probability surface to study the characteristics of the distribution of the input data captured by the network. Ideally, one would like to achieve the best probability surface, best defined in terms of some measure corresponding to a low average error.

During AANN training, the weights of the network are adjusted to minimize the mean square error obtained for each feature vector. If the adjustment of weights is done for all feature vectors once, then the network is said to be trained for one epoch. For successive epochs, the mean square error is averaged over all feature vectors. During testing phase, the features extracted from the test data are given to the trained AANN model to obtain the average error.

### 4.3 Feature Extraction

In this work, the first 19 mel frequency cepstral coefficients, other than the zeroth value, are used. Cepstral mean subtraction is performed to reduce the channel effects. The selected properties for the speech signals are sampling rate of 8 kHz and 16 bit monophonic PCM format. We used a frame rate of 125 frames/sec, where each frame is 16 ms in duration with an overlap of 50 percent between adjacent frames.

### 4.4 The Proposed Speaker Change Detection Algorithm

We begin with the assumption that there is a speaker change located in the data stream at the center of the analysis window under consideration. If the speech signal
of this analysis window comes from different speakers, all the feature vectors in the right half of the window may not fall into the distribution of the feature vectors from the left half window. On the contrary, if the speech signal of this analysis window comes from only one speaker, the feature vectors in the right half of the window falls into the distribution of feature vectors of the left half window.

Given the speech features $\mathcal{S} = \{s_l : l = 1, \ldots, L\}$, $l$ is the frame index and $L$ is the total number of frames in the speech signal. The proposed algorithm for detecting speaker change is summarized as follows:

1. From $L$ frames, $L_w$ number of frames are selected such that $L_w \mod 2 = 1$, and considered as analysis window $w_k$. $w_k$ is the $k^{th}$ analysis window which is given by

$$w_k = \{s_j\}, \quad k \leq j < L_w + k \quad (4.1)$$

2. It is assumed that the speaker change occurs at the middle frame ($c$) of the analysis window.

$$c = k + \left\lfloor \frac{L_w}{2} \right\rfloor \quad (4.2)$$

3. All the frames that are located left of $c$ are considered as left half window $w_l$

$$w_l = \{s_j\}, \quad k \leq j < c - 1 \quad (4.3)$$

Similarly all the frames that are located right of $c$ are considered as right half window $w_r$

$$w_r = \{s_j\}, \quad c + 1 \leq j < L_w + k \quad (4.4)$$

4. AANN is trained using the frames in $w_l$ and the model captures the distribution of this block of data. Then the feature vectors in $w_r$ are given as input to the AANN model and the output of the model is compared with the input to
compute the normalized squared error $e_k$. The normalized squared error $(e_k)$ for the feature vector $x$ is given by

$$e_k = \frac{\|x - o\|^2}{\|x\|^2}$$  \hspace{1cm} (4.5)$$

where $o$ is the output vector given by the model. The error $e_k$ is transformed into a confidence score $s$ using

$$s = \exp(-e_k)$$ \hspace{1cm} (4.6)$$

The average confidence score is calculated by summing the confidence score of the individual frames, and the result is divided by the number of frames in the block. We tried with the weighted summation of the frame scores within the block and there is no improvement in the performance. If true speaker change occurs at $c$, then $w_l$ and $w_r$ will be from different speakers and the average confidence score for this $c$ will be very low. Likewise, if $c$ is not the true speaker change point and both $w_l$ and $w_r$ are from the same speaker, then the average confidence score will be very high. The next possibility is that either $w_l$ or $w_r$ may have the speech features from both the speakers. If this is the case, the average confidence score will be between the above two values.

5. The value of $k$ is incremented by one and the steps from 1 to 4 are repeated until $L_w + k$ reaches $L$.

6. The speaker change points are detected from the confidence score by applying a threshold. The threshold $(t_s)$ is calculated from the confidence score as follows:

$$t_s = s_{\text{min}} (1 + a), \quad 0 < a < 1$$ \hspace{1cm} (4.7)$$

where $s_{\text{min}}$ is the global minimum confidence score and $a$ is the adjustable parameter.
The thresholding step is performed as in any other detection algorithm: the threshold is tuned in accordance to some trade off between false alarms and missed detections. The research community tends to treat false alarms as less cumbersome than missed detections, because over-segmentation caused by a high number of false alarms is easier to remedy than under-segmentation caused by high number of missed detections. In our algorithm, when \( a \) is nearer to 0, the number of missed detections will be more and if it is nearer to 1, the number of false alarms will be more. So the parameter \( a \) is selected to achieve over-segmentation.

### 4.5 Experimental Results

In order to test the performance of the proposed algorithm, several audio records from TV interviews are considered initially. A total dataset of 60 conversations is used in our studies. This includes 20 conversations for each male-male, male-female and female-female speaker conversations. The speaker change points are manually marked. The manual segmentation results are used as the reference for evaluation of the proposed speaker segmentation method. A total of 2,782 speaker segments are marked in the 60 conversations. Excluding the silence, the segment duration is mostly between 0.75 and 5 seconds.

The MFCC feature vectors are extracted for all the speech frames as described in Sections 4.3 and 3.2.1. For each analysis window \((w_k)\), the distribution of the feature vectors is captured using an AANN model as described in Section 4.2. The feature vectors of \( w_l \) are given as input to the AANN and the network is trained for 100 epochs. One epoch of training is a single presentation of all the training vectors to the network. The performance of the AANNs does not change, even if the number of epochs is increased. This is illustrated in Fig. 4.3. There is no significant change
in the performance of the AANN, even though the number of epochs is increased to 1000. Hence, the AANN models are trained for only 100 epochs.

![Graph showing effect of epochs on confidence score](image)

**Fig. 4.3:** Effect of epochs on the confidence score.

The feature vectors of $w_r$ are given as input to the AANN model and the average confidence score is calculated as described in Section 4.4. It is repeated for all analysis window. Fig. 4.4 shows the confidence score obtained for analysis window size of 65, 95, 125 and 140. The number of false alarms and missed detections are significantly low for the 125 frames window size when compared to analysis window size settings of 65, 95 and 140. So, in this work, we used the analysis window size of 125 frames. Moreover, the window size of 125 frames (1 sec.) is appropriate to detect speaker change for short duration speech segments.

The performance of the proposed method for speaker change detection for varying
adjustable parameter is given in Fig. 4.5. It shows that nearly 5.0% missed detections and 5.0% false alarms are achieved for $a=0.52$. The proposed algorithm achieves 84.3%, 94.7% and 89.2% precision, recall and $F$-measure respectively for the TV interviews data.

The performance of this algorithm is compared with support vector machine
Fig. 4.5: Effect of adjustable parameter \((a)\) on the performance of the algorithm.

(SVM) and Gaussian mixture model (GMM) classifiers. The database used for this comparison is the National Institute of Standards and Technology (NIST), Rich Transcription - 2005 Spring Meeting Recognition Evaluation (RT-05S) database [144]. This database is composed of various speech streams recorded during speeches, discussions, and conferences.

In [145], a window scanning approach was proposed using MFCC and SVM classifier. They claimed that SVM training misclassification rate (STMR) is superior to Gaussian training misclassification rate (GTMR), BIC and the commonly used KL2, Mahalanobis, Bhattacharyya and GLR distances. Hence, we compared the performance of our algorithm with SVM classifier and GMM classifier. The feature vector used for AANN classifier is applied for both SVM classifier and GMM classifier. To compute STMR, the linear kernel function with \(\alpha_i=1\) (where \(\alpha_i\) is the user specified positive parameter for the upper bound of the Lagrange multiplier) and STMR threshold of 0.075 are selected. For GMM classifier, we used four Gaussians with GTMR
threshold of 0.04. To compute STMR and GTMR, the feature vectors of the analysis window of size $L_w$, the classifiers are trained and tested for $L_w$ feature vectors. But in our algorithm, the training and testing are done with only $L_w/2$ feature vectors. The performance of the method is compared with STMR and GTMR, and the results are given in Table 4.1.

Fig. 4.6 shows the snapshot of the speaker segmentation using autoassociative neural networks. The left side of the system is allocated for speaker segmentation and diarization. After segmentation, the segmented results are stored in a file as shown in the snapshot.

### 4.6 Summary

An alternate method for speaker segmentation using MFCC features and autoassociative neural network is proposed in this chapter. The average error is used to separate the segments belonging to the two speakers. The performance of the speaker segmentation task depends on the analysis window length and validation threshold. The choice of these parameters is a compromise between a low missed detection rate and a low false alarm rate. The performance of this algorithm has been tested using several real conversations from TV interviews and RT-05S database and also compared with

<table>
<thead>
<tr>
<th>Classifier</th>
<th>$\alpha_r$ (%)</th>
<th>$\beta_r$ (%)</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AANN</td>
<td>15.79</td>
<td>4.68</td>
<td>83.56</td>
<td>95.31</td>
<td>89.05</td>
</tr>
<tr>
<td>SVM</td>
<td>27.01</td>
<td>25.13</td>
<td>66.66</td>
<td>74.87</td>
<td>70.58</td>
</tr>
<tr>
<td>GMM</td>
<td>38.46</td>
<td>37.50</td>
<td>50.12</td>
<td>62.50</td>
<td>55.63</td>
</tr>
</tbody>
</table>
existing algorithms. The results have shown that the proposed approach performs significantly better than the existing metric based approaches. The algorithm can be applied for real time applications and it does not require any prior knowledge about the speaker identity and their model. Moreover, the time taken by the algorithm is less as the AANN model is created by using only one half of the analysis window feature vectors. This work is carried out for only two speaker conversations. The next chapter

Fig. 4.6: Snapshot of the speaker segmentation using AANN.
discusses the application of the method for multi-speaker diarization.