Defence Science Journal, Vol. 60, No. 4, July 2010, pp. 348-363

Temporal Pattern Classification using Kernel Methods for Speech Recognition and Speech Emotion Recognition

C. Chandra Sekhar and S. Chandrakala

Indian Institute of Technology Madras, India.
E-mail: chandra@cse.iitm.ac.in, sckala@gmail.com

ABSTRACT

There are two paradigms for modelling the varying length temporal data namely, modelling the sequences of feature vectors as in the hidden Markov model-based approaches for speech recognition and modelling the sets of feature vectors as in the Gaussian mixture model (GMM)-based approaches for speech emotion recognition. In this paper, the methods using discrete hidden Markov models (DHMMs) in the kernel feature space and string kernel-based SVM classifier for classification of discretised representation of sequence of feature vectors obtained by clustering and vector quantisation in the kernel feature space are presented. The authors then present continuous density hidden Markov models (CDHMMs) in the explicit kernel feature space that use the continuous valued representation of features extracted from the temporal data. The methods for temporal pattern classification by mapping a varying length sequential pattern to a fixed-length sequential pattern and then using an SVM-based classifier for classification are also presented. The task of recognition of spoken letters in E-set, it is possible to build models that use a discretised representation and string kernel SVM based classification and obtain a classification performance better than that of models using the continuous valued representation is demonstrated. For modelling sets of vectors-based representation of temporal data, two approaches in a hybrid framework namely, the score vector-based approach and the segment modelling based approach are presented. In both approaches, a generative model-based method is used to obtain a fixed length pattern representation for a varying length temporal data and then a discriminative model is used for classification. These two approaches are studied for speech emotion recognition task. The segment modelling based approach gives a better performance than the score vector-based approach and the GMM-based classifiers for speech emotion recognition.

Keywords: Hidden Markov model, support vector machine, string kernel, Gaussian mixture model, score vector, parametric vector, speech recognition and speech emotion recognition.

1. INTRODUCTION

Modelling varying length temporal data is important in various domains such as speech recognition, handwritten character recognition, and speaker recognition[1,2]. Temporal data may be of discrete or continuous valued, uniformly or non-uniformly sampled, univariate or multivariate, and of fixed or varying length. Classification of temporal data is a difficult task because the structure of the underlying process has to be inferred and the varying length has to be handled. Two paradigms for modelling the varying length temporal data are: (i) modelling the sequences of vectors and (ii) modelling the sets of vectors. Tasks such as speech recognition need modelling both the temporal dynamics and the correlations among the features in the temporal patterns. In these tasks, production of an example utterance belonging to a class has a fixed number of acoustic events. Hidden Markov models (HMMs) are the commonly used models for speech recognition and the states in an HMM correspond to the acoustic events[1,3]. In tasks such as speaker recognition, spoken language identification, audio classification, music classification and speech emotion recognition, the length of utterance is large and the local temporal dynamics is not critical. Production of different examples of a class may have different number of acoustic events. Gaussian mixture model (GMM)-based approaches that model the sets of feature vectors in the temporal data are used for such tasks[4-7].

Generative approaches[1-3,7-10] and discriminative approaches[11-15] are two main approaches to design classifiers. Generative approaches rely on a learned model of the joint probability distribution of the observed data and the corresponding class membership. These approaches are not suitable for classifying the data of confusable classes[16] because a model is built for each class using the data belonging to that class only. Kernel methods to develop discriminative methods for classification of varying length temporal patterns has been considered. Building hidden Markov models in the kernel feature space has been explored. Kernel-based clustering method[17] is used to perform vector quantisation in the kernel feature space and obtain a sequence of codebook indices from the sequence of vectors in a temporal pattern. These sequences of codebook indices

Received 28 September 2009, Revised 26 March 2010

are used to build the discrete HMMs in the kernel feature space. Classification of temporal patterns represented by sequences of codebook indices using a string kernel-based support vector machine (SVM) has also been explored. For building continuous density HMMs in the kernel feature space, the explicit kernels have been considered for which the kernel feature space is explicitly defined.

In the second category of approaches to classification of temporal patterns represented as sequence of vectors, the authors explore the methods for obtaining a fixed-length sequential pattern from a varying length temporal data have been exploredand then classifying the fixed-length sequential pattern using an SVM-based classifier. For modelling sets of vectors based representation of temporal data, two approaches in a hybrid framework are considered. In these approaches, a generative model-based method is used to obtain a fixed dimensional vector representation for a varying-length temporal data and then an SVM is used for classification.

2. KERNEL METHOD FOR CLASSIFICATION AND CLUSTERING

A kernel method comprises two steps[18]: a nonlinear transformation, that maps data points from a low-dimensional input space to a high-dimensional feature space induced by an innerproduct kernel or a Mercer kernel, and a learning algorithm to find optimal linear solutions in that feature space. For nonlinearly separable data of two classes in the input space, the nonlinear transformation may lead to linear separability or linear nonseparability of the data in the kernel feature space.

2.1 Support Vector Machine

The SVM constructs an optimal hyperplane by finding the maximum margin hyperplane as a decision surface to separate the positive and negative examples of a class in the kernel feature space. The margin is the distance of the nearest example to a hyperplane. Two linearly separable classes in the kernel feature space, as illustrated in Fig.1 are considered. A hyperplane is specified by $(\bar{w},b)$ where $\bar{w}$ is the weight parameter vector and b is the bias. The margin of a hyperplane is given by $\frac{1}{\|\bar{w}\|}$. Maximising the margin is equivalent to minimising the Euclidean norm of the weight vector.

Let the set of training examples and their class labels be $\{(\bar{x}_i, y_i)\}, i = 1,2,...,N$. The separating hyperplane must satisfy the constraints:

$$y_i(\bar{w}^T\bar{z}_i + b) \geq 1, i = 1,2,...,N$$

where $\bar{z}_i$ is the feature vector for the input space vector $\bar{x}_i$ and $y_i \in \{+1,-1\}$ is the corresponding class label. The cost function is given by

$$\psi(\bar{w}) = \frac{1}{2}\bar{w}^T\bar{w}$$

The objective function for the constrained optimisation problem is given by:

$$L_p(\bar{w},b,\bar{\lambda}) = \frac{1}{2}\bar{w}^T\bar{w} - \sum_{i=1}^{N}\lambda_i[y_i(\bar{w}^T\bar{z}_i + b) - 1]$$

where λ_i are the Lagrange multipliers. Conditions for optimality are given as:

$$\frac{\partial L_p(\bar{w},b,\bar{\lambda})}{\partial \bar{w}} = \bar{0}$$

$$\frac{\partial L_p(\bar{w},b,\bar{\lambda})}{\partial b} = \bar{0}$$

Application of optimality conditions gives

$$\bar{w} = \sum_{i=1}^{N}\lambda_i y_i \bar{z}_i$$

$$\sum_{i=1}^{N}\lambda_i y_i = 0$$

Substituting these optimality conditions gives the dual form of the objective function as

$$L_d(\bar{\lambda}) = \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j(\bar{z}_i^T\bar{z}_j)$$

The Lagrange multipliers $\{\lambda_i\}$ maximise the above objective function subject to the constraints

$$\sum_{i=1}^{N}\lambda_i y_i = 0$$

$$\lambda_i \geq 0, i = 1,2,...,N$$

With the Lagrangian optimisation done, the data points for which Lagrange multipliers are not zero are the support vectors. The support vectors are the small subset of training data-set that lie on the margin as illustrated in Fig 1. For the optimum Lagrange multipliers $\{\lambda_i^*\}$, the optimum weight vector $\bar{w}$ is given by

$$\bar{w} = \sum_{i=1}^{N_s}\lambda_i^* y_i \bar{z}_i$$

where N_s is the number of support vectors.

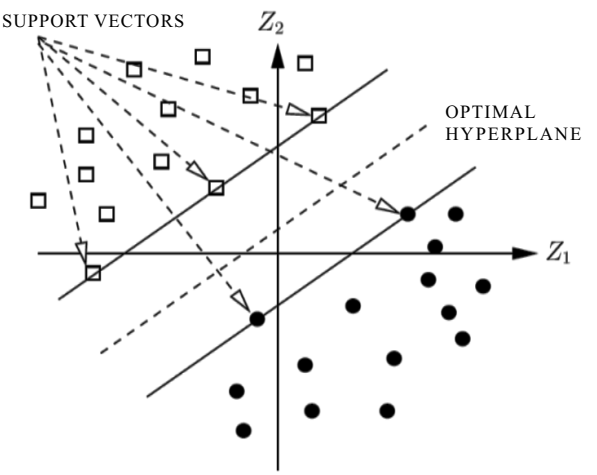

Figure 1. Linearly separable classes.

Now we consider two linearly nonseparable classes in the kernel feature space as illustrated in Fig. 2. It is seen that some data points may fall inside the region of separation or on the wrong side of separation. Let ξ_i is a measure of the deviation for $\bar{z}_i$ from the ideal condition of pattern separability. Here, the learning involves in finding the values for $\bar{w}$ and b such that they satisfy the following constraints:

$$y_i(\bar{w}^T\bar{z}_i + b) \geq 1 - \xi_i, i = 1, 2, ..., N$$

$$\xi_i \geq 0, i = 1, 2, ..., N$$

The cost function is given by

$$\psi(\bar{w}) = \frac{1}{2}\bar{w}^T\bar{w} + C\sum_{i=1}^{N}\xi_i$$

where C is the specified parameter that assigns a penalty to the misclassification. Now, similar to the case of linearly separable classes, the problem becomes finding the Lagrange multipliers $\{\lambda_i\}$ that maximise the following dual form of objective function

$$L_d(\bar{\lambda}) = \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j(\bar{z}_i^T\bar{z}_j)$$

subject to the constraints

$$\sum_{i=1}^{N}\lambda_i y_i = 0$$

$$0 \leq \lambda_i \leq C, i = 1, 2, ..., N$$

For the optimum Lagrange multipliers $\{\lambda_i^*\}$, the optimum weight vector $\bar{w}$ is given by

$$\bar{w} = \sum_{i=1}^{N_s}\lambda_i^* y_i\bar{z}_i$$

where N_s is the number of support vectors. The discriminant function for the optimal hyperplane in terms of the support vectors is defined as

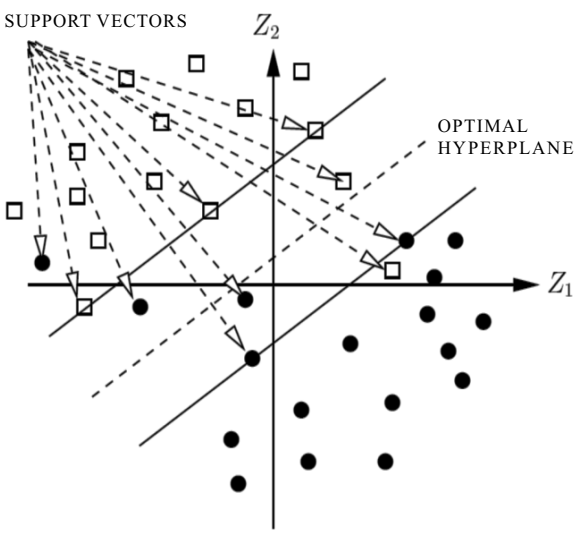

Figure 2. Linearly nonseparable classes.

$$\bar{w}^T\bar{z} + b = \sum_{i=1}^{N_s}\lambda_i^* y_i\bar{z}^T\bar{z}_i + b$$

The notion that is central to the construction of the support vector learning algorithm is the innerproduct kernel operation between a support vector $\bar{x}_i$ and an input vector $\bar{x}$. Innerproduct kernel $K(\bar{x}, \bar{x}_i)$ is defined as $K(\bar{x}, \bar{x}_i) = \bar{z}^T\bar{z}_i$. The various innerproduct kernel functions used are polynomial, sigmoidal and Gaussian kernels.

In the statistical methods for pattern classification, as the dimension of the pattern vector increases, the number of parameters to be estimated also increases. For neural networks, an increase in the dimension of the pattern vector increases the number of input nodes, and hence, it increases the number of weight parameters to be estimated. So, they need a large number of examples for proper training. The number of examples required to build an SVM is not dependent on the dimension of pattern vector. Kernel method for nonlinear separation of clusters of data is discussed in the next section.

2.2 Clustering in Kernel Feature Space

The commonly used K-means clustering method gives a linear separation of data as illustrated in Fig. 3 and is not suitable for separation of nonlinearly separable data. In this subsection, the criterion for partitioning the data into clusters in the input space using the K-means clustering algorithm is first presented. Clustering in the kernel feature space is then realised using the K-means clustering algorithm[17,19].

Consider a set of N data points in the input space, $\bar{x}_i, i = 1, 2, ..., N$. Let the number of clusters to be formed is K. The commonly used criterion for partitioning of the data into K clusters is to minimise the trace of the within-cluster scatter matrix, S_w, defined as follows[17]:

$$S_w = \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}(\bar{x}_i - \bar{m}_k)(\bar{x}_i - \bar{m}_k)^T$$

where $\bar{m}_k$ is the centre of the k^{th} cluster, C_k, and γ_{ki} is the membership of data point $\bar{x}_i$ to the cluster C_k. The membership value $\gamma_{ki} = 1$ if $\bar{x}_i \in C_k$ and 0 otherwise. The number of points in the k^{th} cluster is given as N_k defined by:

$$N_k = \sum_{i=1}^{N}\gamma_{ki}$$

The centre of the cluster C_k is given as $\bar{m}_k$ defined by:

$$\bar{m}_k = \frac{1}{N_k}\sum_{i=1}^{N}\gamma_{ki}\bar{x}_i$$

The optimal partitioning of the data points involves determining the indicator matrix, Γ, with the elements as γ_{ki}, that minimises the trace of the matrix S_w. This method is used in the K-means clustering algorithm for linear separation of the clusters. For nonlinear separation of clusters of data points, the input space is transformed

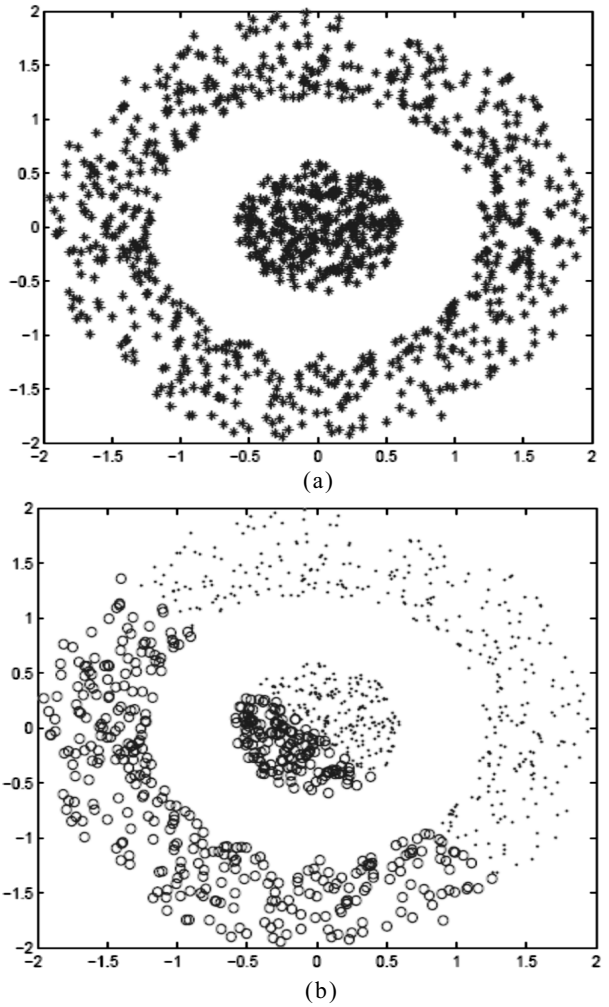

Figure 3. (a) Scatter plot of the data in clusters separable by a circular shaped curve in a 2-dimensional space. Inner cluster belongs to class 1 and the outer cluster belongs to class 2. (b) Linear separation of data obtained using K-means clustering in the input space.

into a high-dimensional feature space using a smooth and continuous nonlinear mapping, ϕ, and the clusters are formed in the feature space. The optimal partitioning in the feature space is based on the criterion of minimising the trace of the within-cluster scatter matrix in the feature space, S_w^{ϕ}. The feature space scatter matrix is given by:

$$S_w^{\phi} = \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}(\phi(\bar{x}_i)-\bar{m}_k^{\phi})(\phi(\bar{x}_i)-\bar{m}_k^{\phi})^T$$

where $\bar{m}_k^{\phi}$, the centre of the k^{th} cluster in the feature space is given by

$$\bar{m}_k^{\phi} = \frac{1}{N_k}\sum_{i=1}^{N}\gamma_{ki}\phi(\bar{x}_i)$$

The trace of S_w^{ϕ} can be computed using the innerproduct operations as given below:

$$Tr(S_w^{\phi}) = \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}(\phi(\bar{x}_i)-\bar{m}_k^{\phi})^T(\phi(\bar{x}_i)-\bar{m}_k^{\phi})$$

When the feature space is explicitly represented, as in the case of mapping using polynomial kernels, the K-means clustering algorithm can be used to minimise the trace given in the above equation. However, for Mercer kernels with implicit mapping used for transformation, it is necessary to express the trace in terms of kernel function. The Mercer kernel function in the input space corresponds to the inner-product operation in the feature space, i.e.,

$$\mathrm{K}_{ij} = \mathrm{K}(\bar{x}_i,\bar{x}_j) = \phi^T(\bar{x}_i)\phi(\bar{x}_j)$$

The trace of S_w^{ϕ} can be rewritten as

$$Tr(S_w^{\phi}) = \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}\mathrm{K}_{ii} - \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}\sum_{j=1}^{N}\frac{\gamma_{kj}}{N_k}\mathrm{K}_{ij}$$

$$= \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}[\mathrm{K}_{ii} - \frac{1}{N_k}\sum_{j=1}^{N}\gamma_{kj}\mathrm{K}_{ij}\ ]$$

$$= \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\gamma_{ki}D_{ki}$$

where

$$D_{ki} = \mathrm{K}_{ii} - \frac{1}{N_k}\sum_{j=1}^{N}\gamma_{kj}\mathrm{K}_{ij}$$

The term D_{ki} is the penalty associated with assigning $\bar{x}_i$ to the k^{th} cluster in the feature space. For explicit mapping kernels such as the polynomial kernel function, the feature space representation is explicitly known. Polynomial kernel is given by $\mathrm{K}(\bar{x},\bar{x}_i) = (a\bar{x}^T\bar{x}_i + c)^g$, where a and c are constants, and d is the degree of polynomial kernel. The vector $\phi(\bar{x})$ in the feature space of the polynomial kernel corresponding to the input space vector $\bar{x}$ includes the monomials upto order d of elements in $\bar{x}$. For a polynomial kernel, D_{ki} may take a negative value because the magnitude of K_{ij} can be greater than that of K_{ii}. To avoid D_{ki} taking negative values, K_{ij}, in the equation for D_{ki} is replaced with the normalised value $\hat{\mathrm{K}}_{ij}$ defined as

$$\hat{\mathrm{K}}_{ij} = \frac{|\mathrm{K}_{ij}|}{\sqrt{\mathrm{K}_{ii}}\sqrt{\mathrm{K}_{jj}}}$$

From Cauchy-Schwarz inequality, $\mathrm{K}_{ij} \le \sqrt{\mathrm{K}_{ii}}\sqrt{\mathrm{K}_{jj}}$. It follows that for the polynomial kernel $\hat{\mathrm{K}}_{ii} = 1$ and $\hat{\mathrm{K}}_{ij} \le \hat{\mathrm{K}}_{ii}$, and D_{ki} is defined as:

$$D_{ki} = \hat{\mathrm{K}}_{ii} - \frac{1}{N_k}\sum_{j=1}^{N}\gamma_{kj}\hat{\mathrm{K}}_{ij}$$

For implicit mapping kernels such as the Gaussian kernel function, the explicit feature space representation is not known. A Gaussian kernel is defined as $\mathrm{K}(\bar{x},\bar{x}_i) = \exp(-\frac{\|\bar{x}-\bar{x}_i\|^2}{\sigma})$ where σ is the kernel width parameter. For Gaussian kernel, D_{ki} takes a nonnegative value because $\mathrm{K}_{ii} = 1$ and $\mathrm{K}_{ij} \le \mathrm{K}_{ii}$. Now, the optimisation problem is to determine the indicator matrix such that

$$\Gamma^* = \arg_{\Gamma}\min Tr(S_w^{\phi})$$

An iterative method for solving this optimisation problem is given in[17]. The clusters obtained for the ring data using the kernel based clustering method are shown in Fig. 4. The kernel based clustering method is used for construction of discrete HMMs in the kernel feature space as explained in the next section.

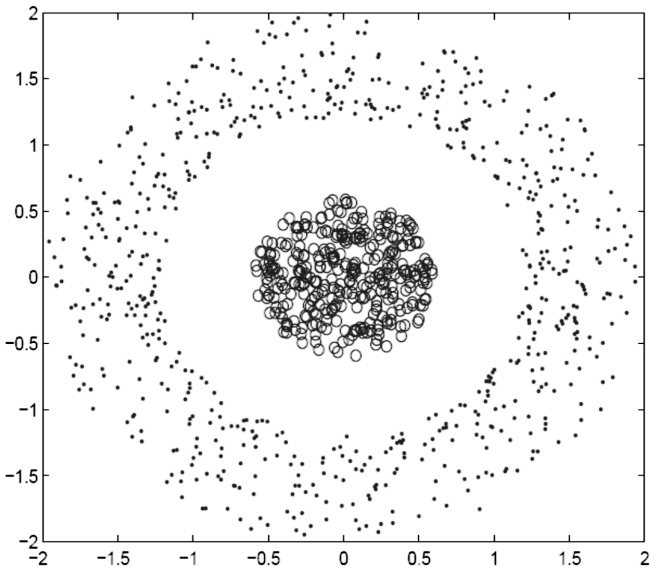

Figure 4. Nonlinear separation of data obtained using kernel-based clustering in feature space for the data plotted in Fig. 3

3. CLASSIFICATION OF VARYING LENGTH TEMPORAL PATTERNS REPRESENTED AS SEQUENCES OF DISCRETE SYMBOLS

One of the problems that arise in the temporal pattern classification is the variation in the durations of sequences. In case of tasks such as speech recognition, the durations of two different utterances of a given word, uttered by the same speaker, will be different. The variation in the durations of sound units leads to variation in the length of the sequence of feature vectors. The hidden Markov models (HMMs) are successfully used for temporal data classification. A method of construction of HMMs in the kernel feature space, where the confusable patterns are more easily separable, is proposed.

3.1 Hidden Markov Models in the Input Space

Hidden Markov models have been extensively used for modelling of varying-length temporal data. In speech recognition task, the HMM for a class is trained using the varying length observation sequences corresponding to the sequences of speech feature vectors extracted from the speech signal data of multiple examples of the class. The HMM for a class is trained to maximise the likelihood of the model generating the observation sequences of that class. During recognition, the observation sequence of a test pattern is given as input to the HMM of each class, to compute the probability of the test sequence being generated by that model. Then the class of the model with the highest probability is assigned to the test pattern. A discrete HMM is a finite state machine characterised by the number of states in the model, the state-transition probability distribution, the observation symbol probability distribution for each state, and the initial state probability distribution. Continuous density HMMs (CDHMMs) use probability densities to represent the continuous observation distributions of the states. The continuous observation density for a state is estimated by assuming that it can be represented by a mixture of Gaussian density functions. Then the estimation of continuous density for a state involves estimation of the mean vector and covariance matrix of each component of the Gaussian mixture, and the estimation of the mixture coefficients. Therefore, construction of CDHMMs in the input space involves probability density estimation in the input space. In a discrete HMM, the observation symbols are discrete symbols and a discrete probability distribution is used to represent the observation symbol probability distribution of each state. The continuous signal representation of temporal data is converted into discrete symbols using vector quantisation. The codebook used for vector quantisation is built by clustering the data vectors of all the classes and including the mean vectors of the clusters as codevectors in the codebook. Construction of discrete HMMs in kernel feature space where the confusable patterns are more easily separable is proposed.

3.2 Discrete Hidden Markov Models in Kernel Feature Space

Construction of discrete hidden Markov models (DHMM) for classification of varying length patterns in a multidimensional input space involves clustering and vector quantisation of multidimensional vectors. Proper clustering of nonlinearly separable data is important for clustering-based approaches to pattern classification. The commonly used *K*-means clustering method gives a linear separation of data and is not suitable for separation of nonlinearly separable data. The kernel-based clustering technique involves transformation of nonlinearly separable data into a high-dimensional feature space induced by the inner-product kernels or Mercer kernels where the data is expected to be linearly separable. Then, clustering is performed in the high-dimensional kernel feature space. It is possible to construct the discrete hidden Markov models (DHMMs) in the polynomial kernel feature space, by clustering and vector quantisation of the explicit feature vectors corresponding to the input space vectors. However, for construction of DHMMs in the feature spaces of kernels that perform implicit mapping, it is necessary to perform clustering and vector quantisation in the feature space using kernel functions only. In the proposed method, we have employed a kernel based clustering method proposed in[17] and explained in the previous section.

In vector quantisation, an input data vector $\bar{x}$ is assigned the index of the cluster whose center has the

highest similarity to $\bar{x}$. For a codebook constructed in the input space or in the polynomial kernel feature space, the code vectors correspond to the mean vectors of the clusters. For the clusters formed in the implicit kernel feature space, there is no explicit representation of the mean vector of a cluster. The similarity measure between an input vector and the center of a cluster, C_k, in the feature space can be computed as follows:

$$s_k(\bar{x}) = \phi^T(\bar{x}) m_k^{\phi} = \frac{1}{N_k} \sum_{\bar{x}_i \in C_k} \mathrm{K}(\bar{x}, \bar{x}_i)$$

Computation of $s_k(\bar{x}), k = 1, 2, ..., K$ involves performing the kernel operation between $\bar{x}$ and each of the N data vectors used in formation of the clusters. Therefore vector quantisation in the kernel feature space is computationally intensive. To reduce the number of kernel operations, a method in which a cluster in the feature space is represented by a subset of its vectors close to its center is proposed. The similarity measure in the above equation is applied to each vector in a cluster to compute its similarity to the centre of the cluster. The vectors with high similarity are considered to be close to the center of the cluster. Let S be the size of the subset of vectors close to the centre of a cluster used to represent the cluster. If the mean vector of S vectors is approximately the same as the mean vector of all the data vectors in a cluster, then the cluster can be represented by the S vectors in the subset for the purpose of vector quantisation. The similarity measures $s_k(\bar{x}), k = 1, 2, ..., K$ are expected to be close to the similarity measures of $\bar{x}$ to the mean vectors of S vectors in each cluster. For $S \ll N_k$, this method leads to a significant reduction in the computational complexity of vector quantisation in kernel feature space. Figure 5 illustrates the method for a cluster. The 10 close points to the centre of the cluster are shown inside the inner circle and the 50 close points are shown inside the outer circle.

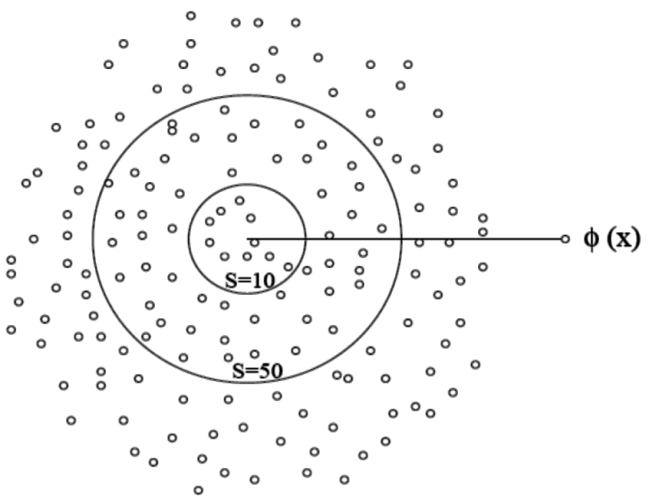

Figure 5. Illustration of representation of a cluster by *S* points close to its centre for different values of *S*.

Generative models such as hidden Markov model constructed in the kernel feature space still rely on a learned model of the joint probability distribution of the observed data and the corresponding class membership. These approaches are not suitable for classifying the data of confusable classes[16] because a model is built for each class using the data belonging to that class only. The string kernel-based SVM for classification of sequence of codebook indices representing a temporal pattern is discussed.

3.3 String Kernel-based SVM

The kernel-based classification using a sequence kernel is a good choice to classify the temporal data of varying length which can be represented in the form of sequences of symbols. Shengfeng[20], *et al.* have grouped the sequence kernels into two categories based on the way the similarity between two sequences is defined. One way to define a sequence kernel is based on the distance between two sequences. Edit distance kernel and modified edit distance kernel belong to this category. The other way is based on the number of common subsequences in the two input sequences. String kernel[21], mismatch kernel[22] and their variants fall into this category. The string kernel is shown to give a better performance than the polynomial kernel or Gaussian kernel for text classification[21].

In this work, we use the string kernel-based SVM classifier for classification of a sequence of codebook indices obtained using clustering and vector quantisation in the kernel feature space. The string kernel computes the similarity between two strings of symbols using the subsequences present in the sequences and their positions. The more the number of subsequences common to them, the more similar the two strings are. A subsequence is an ordered sequence of p symbols in a string. Symbols in a subsequence do not need to be contiguous. To deal with non-contiguous subsequences, a decay factor $\lambda \in (0,1)$ is introduced. Let Σ be a finite alphabet of symbols and a string is a finite sequence of symbols from Σ, including the empty sequence. A string s is denoted by $s = s_1, ..., s_{|s|}$ where $|s|$ is the length of the string . Then, u is a subsequence of s, if there exist indices $i = (i_1, ..., i_{|u|})$, with $1 \le i_1 \le ... \le i_{|u|} \le |s|$, such that $u_j = s_{i_j}$, for $j = 1, ..., |u|$, or $u = s[i]$ for short. The length $l(i)$of the subsequence in s is $i_{|u|} - i_1 + 1$. The set of all subsequences of length n is denoted by Σ^n. The feature mapping ϕ for a string s is given by defining $\phi_u(s)$ for each $u \in \Sigma^n$ as follows:

$$\phi_u(s) = \sum_{i:u=s[i]} \lambda^{l(i)}$$

where $\lambda \le 1$. These features measure the number of occurrences of subsequences in the string s weighting them according to their lengths. The innerproduct of the feature vectors of the two strings s and t gives a sum over all common subsequences weighted according to their frequency of occurrence and lengths as follows:

$$\mathrm{K}(s,t) = \sum_{u \in \Sigma^n} \sum_{i:u=s[i]} \sum_{j:u=t[j]} \lambda^{l(i)+l(j)}$$

where j is the index sequence for the symbol subsequence u in string t. The values of n=3 and λ=0.01 and are empirically chosen in this study.

3.4 Classification of Temporal Patterns Represented as Sequences of Symbols

3.4.1 Description of Database

The performance of the proposed approaches for a task in speech recognition has been studied. The task involves recognition of spoken utterances of a highly confusable subset of letters in English alphabet, namely, E-set. The E-set includes the following 9 letters: {B, C, D, E, G, P, T, V, Z}. The OGI spoken letter database[23] is used in the study on recognition of E-set. The training data set consists of 240 utterances for each letter, and the test data set consists of 60 utterances per letter. For the speech signal of each utterance, short-time analysis of speech is performed using a frame size of 25 ms with a shift of 10 ms resulting in a sequence of data vectors. Each data vector consists of 12 Mel-frequency cepstral coefficients (MFCC), energy, their first-order derivatives (delta coefficients) and their second-order derivatives (acceleration coefficients) resulting in a dimension of 39. The number of data vectors in the training data set of each class is about 12,000.

3.4.2 DHMMs in the Kernel Feature Space

Recognition of E-set, a 5-state, left-to-right, discrete HMM is constructed for each class (letter) in this study. For input space DHMMs, a codebook of size 64 is constructed in the 39-dimensional space by clustering the data vectors of all the classes into 64 clusters using K-means clustering algorithm. For construction of DHMMs in the feature space of polynomial kernel, the explicit clustering method and the implicit clustering method have been considered. In the explicit clustering method, a codebook of size 64 is constructed in the explicitly represented feature space of polynomial kernel of degree 2. The feature vector includes the monomials of order 0,1 and 2 derived from the 39-dimensional input space vector. Therefore the dimension of the feature space is 820. The K-means clustering algorithm is used for explicit clustering. In the implicit clustering method, the stochastic method is used for determining the elements of the indicator matrix.

In the all-class-data clustering method, the clustering is done for the data of all the classes. However, only the data vectors of 30 utterances in the training set of a letter are used in clustering. This is mainly to ensure that the stochastic method converges and that the computational complexity of clustering does not become high. A codebook of size 64 is constructed from the data of all the classes. In the class-wise-data clustering method, the data vectors belonging to the total training set of a letter are used in building a codebook of size 8, i.e., 8 clusters are formed from the data vectors of a class. The total number of clusters for all the 9 letters in the E-set is 72. Implicit clustering using the all-class-data clustering and the class-wise-data clustering methods is carried out for the polynomial kernel of degree 2 and for the Gaussian kernel with a σ value of 500. After the codebook is constructed using a particular method, the DHMMs are built by performing vector quantisation using the codebook. For the codebooks constructed using the explicit clustering method, vector quantisation is done using all the data points of a cluster in computing the similarity of a data vector with the mean of the cluster, i.e., $S = N_k$ is used.

The classification performance on the test data set for DHMMs built using different methods of kernel-based clustering and for different kernels is given in Table 1. The performance is compared with that of the input space DHMMs. It is seen that the explicit clustering in the feature space of polynomial kernel gives a marginally better performance. The poorer performance of the all-class-data clustering method may be because only a subset of training data set is used in building the codebook. The performance of class-wise-data clustering method for Gaussian kernel gives a significantly better performance compared to the input space HMMs. It is interesting to note that the performance of the implicit clustering method for Gaussian kernel is better than that of the implicit clustering method for the polynomial kernel.

Table 1. Classification accuracy of DHMMs in the input space and the kernel feature space for E-set recognition task

Models	Clustering method	Accuracy (per cent)
DHMMs in the input space	K -means clustering	65.1
DHMMs in polynomial kernel space	Explicit, all-class-data clustering	67.96
	Implicit, all-class-data clustering	66.11
	Implicit, class-wise-data clustering	63.15
DHMMs in Gaussian kernel space	Implicit, all-class-data clustering	65.19
	Implicit, class-wise-data clustering	70.74

3.4.3 Studies on String Kernel-based SVM Classifier

The performance of the string kernel-based SVMs using the sequence of codebook indices obtained by performing VQ in kernel feature space is shown in Table 2. The performance of the string kernel-based SVM using the sequence of codebook indices obtained by performing VQ in the Gaussian kernel feature space is higher by about 13 per cent compared to the performance of DHMMs using the sequence of codebook indices obtained by performing VQ in the input space.

These studies demonstrate that discrete HMMs in

Table 2. Classification accuracy (in per cent) of string kernel-based SVM models using the sequence of codebook indices for E-set recognition task

Method for vector quantisation	Classification model	Accuracy (per cent)
VQ in the Input space	DHMMs	65.19
	SVM using string kernel	82.41
VQ in polynomial kernel feature space	DHMMs	67.96
	SVM using string kernel	88.89
VQ in the Gaussian kernel feature space	DHMMs	70.74
	SVM using string kernel	95.55

the kernel feature space give an improved performance over the discrete HMMs in the input space. Classification of sequence of codebook indices obtained using VQ in the kernel feature space, with a string kernel based SVM gives a significantly better performance than the discrete HMMs in the kernel feature space.

4. CLASSIFICATION OF VARYING LENGTH TEMPORAL PATTERNS REPRESENTED AS SEQUENCES OF FEATURE VECTORS

4.1 CDHMMs in the Explicit Kernel Feature Space

It is important to note that the performance of DHMMs in the kernel feature space is not expected to be as good as the performance of the CDHMMs in the input space. This is mainly due to the significant loss of information incurred in discretisation of continuous signal representations using vector quantisation in construction of the DHMMs. Performance of the CDHMMs in the kernel feature space is expected to be better than that of the CDHMMs in the input space. Construction of a CDHMM in the input space involves estimation of initial state probabilities, state transition probabilities, and continuous observation probability density functions for each state in the CDHMM. For construction of CDHMMs in the feature space of an explicit mapping kernel, the methods used for construction of the CDHMMs in the input space can be used. For explicit mapping kernels such as the polynomial kernel function, the feature space representation is explicitly known. The polynomial kernel is defined by:

$$\mathrm{K}_{ij} = \mathrm{K}(\bar{x}_i, \bar{x}_j) = (a\bar{x}_i^T \bar{x}_j + c)^g$$

where g is the degree of the polynomial kernel and a and c are constants. The vector $\phi(\bar{x})$ in the feature space of the polynomial kernel corresponding to the input space vector $\bar{x}$ includes the monomials upto order g of elements in $\bar{x}$. For a d-dimensional input space vector, the dimension D of the vector in the feature space of polynomial kernel of degree g is given by:

$$D = \frac{(d+g)!}{d!g!}$$

However, as the dimension of the feature space is high, construction of CDHMMs in the feature space would have a significantly higher computational complexity and would need larger training data sets for proper estimation of the parameters of the HMMs.

4.2. Conversion of Varying-length Sequences of Feature Vectors to Fixed-length Sequential Patterns

Generally, in acoustic modelling using SVMs, the varying-length sequence of feature vectors is mapped to a fixed-length pattern. In the split and average method, the speech signal can be assumed to be homogeneous in nature for context independent subword units such as monophones. Hence the segment of a monophone is divided into a fixed number of parts, and each part is represented by the central frame of that part or by the average of frames in that part[24,25]. This method is not suitable for context-dependent units such as diphones, triphones, syllables and words that are not homogeneous in nature. One of the methods considered for context-dependent units is the deletion and replication of least varying frames. In this method[26], the frame in a segment that has the minimum distance from its adjacent frames is identified. Depending on whether the segment length is greater than the chosen pattern length, the frame with the minimum distance is removed from the segment or replicated in the segment. This process is continued until the length of the segment is the same as the chosen length of the pattern. The performance of this method depends on the distance measure used to identify the least varying frames. For SVM-based classifiers, we consider four approaches to obtain the fixed-length patterns namely, the linear compaction and elongation method, anchor point-based method, outerproduct matrix method and the varying frame-shift method.

4.2.1 Linear Compaction and Elongation

In this method[27], if the number of frames in a segment is greater than the fixed length, a few frames are omitted. If the number of frames in a segment is smaller than the fixed length, a few frames are repeated. Consider a segment with L_s frames in it. For a chosen pattern length L_p, the linear relationship between the index s of a frame in the segment and the index p of a frame in the pattern is given by

$$s = (\frac{p * L_s}{L_p})$$

A limitation of this method is that omission of frames may result in abrupt discontinuities and loss of frames that may be important for pattern analysis.

4.2.2 Anchor Point-based Method

For Consonant-Vowel (CV) units of speech, the important information necessary for pattern analysis is present in the region around the vowel onset point. The point at which the consonant ends and the vowel begins in a CV segment is defined as the vowel onset point (VOP). A

multilayer perceptron can be trained to detect the VOPs in CV utterances automatically[28]. Then a fixed-length region around the VOP is considered for analysis[29]. This segment is analysed to obtain a fixed-length pattern. In this method, no crucial frame will be lost, but it is not suitable for context-dependent units of other types.

4.2.3 Outerproduct Matrix Method

In this method[30], a fixed dimensional pattern vector is derived from a sequence of multidimensional vectors (trajectory) by considering it as a matrix and then by carrying out the outerproduct operation on the trajectory matrix. Short-time analysis of a speech segment gives a sequence of *l*-dimensional vectors that is considered as a trajectory in the *l*-dimensional space. Let *m* be the number of frames for a given segment. The trajectory matrix for the segment consists of the *m* frames, $\bar{x}_1, \bar{x}_2, ..., \bar{x}_m$, as its columns. The trajectory matrix of *l*-by-*m* trajectory matrix, *X*, is given as :

$$X = [\bar{x}_1, \bar{x}_2, ..., \bar{x}_m]$$

The value of *m* will be different for speech segments of different durations. The outerproduct matrix, *Z*, of a trajectory matrix *X* is given by:

$$Z = XX^T$$

The outerproduct matrix *Z* is an *l*-by-*l* matrix. The dimension of the outerproduct matrix is independent of the number of frames in the trajectory. The outerproduct matrix is vectorized to obtain a fixed dimension pattern that can be used as input to SVM based classifier. However, for an utterance consisting of multiple sounds, the outerproduct operation on the trajectory matrix of its speech segment leads to an averaging operation over non-homogeneous segments of different sounds resulting in the possible loss of important discriminatory information necessary for recognition of sounds.

A variant of this method, named multiple outerproduct matrices method, is proposed in which the outerproduct matrix is computed for the trajectory matrix of the segment of each sound in an utterance, and then the fixed dimension pattern is derived from the outerproduct matrices of the different sounds in the utterance as shown in Fig. 6. This method requires segmentation of the speech signal of an utterance into the regions of different sounds in it. A pattern extracted using the outerproduct matrix method captures only the correlation. Another variant of outerproduct matrix method, namely the augmented pattern extraction method has been proposed in which the outerproduct matrix is computed for the trajectory matrix of the segment of each sound in an utterance, and then the fixed dimension pattern is derived from the outerproduct matrices of the different sounds in the utterance augmented with a fixed number of frames around the VOP.

4.2.4 Varying-frame Rate Method

In the varying frame rate method, the frame rate is varied for each utterance to obtain a predefined number of frames from the speech signal of an utterance. The frame shift for each utterance is chosen based on the duration of the utterance. For an utterance consisting of multiple sounds (phonemes), the varying-frame rate method may lead to loss of important discriminatory information necessary for recognition of sounds. Therefore, we the multiple varying frame rates method has been proposed, in which the varying-frame rate method is used independently for each sound unit in an utterance. This method requires segmentation of the speech signal of an utterance into the regions of different sounds in it. In this study, only the consonant-vowel (CV) type units were considered. For deriving the fixed-dimension pattern vector using the multiple varying frame rates method, the vowel onset point (VOP) was detected using the approach given in[28]. The segment of the utterance before the VOP corresponds to

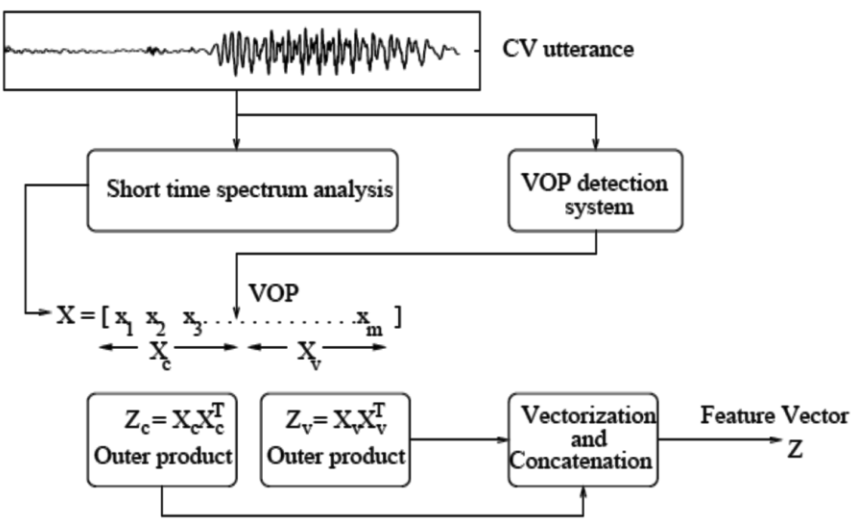

Figure 6. Block diagram for multiple outerproduct matrices method.

the consonant, and the segment after the VOP corresponds to vowel. The varying frame rate method extracts N_c frames for the consonant region and N_v frames for the vowel region. The value of the frame shift for the consonant region of duration T_c is determined as T_c / N_c. The value of the frame shift for the vowel region of duration T_v is determined as T_v / N_v. Thus the frame shift may be different for different sounds in an utterance. Using the multiple varying frame rates method, a fixed length pattern for a speech utterance is obtained by concatenating the N_c and N_v frames.

4.3 Classification of Temporal Patterns Represented as Sequences of Feature Vectors

4.3.1 CDHMM in Polynomial Kernel Feature Space

The polynomial kernel of degree 2 is used in this study. The 820 dimensional feature vector includes the monomials of order 0, 1 and 2 derived from the 39-dimensional input space data vector. A 5-state, left-to-right HMM is constructed for each class. The CDHMMs in the input space use two mixtures per state. The CDHMMs in the polynomial kernel feature space use one mixture per state. The performance of different approaches to build HMMs is given in Table 3. The performance of DHMMs in the input space and in the polynomial kernel feature space is inferior (by 15 per cent) to that of the performance of the CDHMMs in the input space. However, the performance of the CDHMMs in the kernel feature space is better than that of the CDHMMs in the input space (by about 10 per cent).

Table 3. Classification accuracy of DHMMs and CDHMMs in the input space and in the polynomial kernel feature space for E-set recognition task

Classification model	Accuracy (per cent)
DHMMs in the input space	65.19
DHMMs in the polynomial kernel feature space	67.96
CDHMMs in the input space	82.96
CDHMMs in the polynomial kernel feature space	93.24

4.3.2 Classification of Fixed Length Patterns Derived from Varying Length Sequences

The performance of the SVM models built using the fixed length patterns based on the proposed approaches is given in Table 4. For linear compaction and elongation method, a frame shift of 10 milliseconds is used. The number of frames is chosen as 58 (obtained from the average duration of all utterances) for the linear compaction and elongation method, 40 for the varying frame rate method, and 15 frames in the consonant region and 25 frames in the vowel region for the multiple varying frame rates methods respectively. The performance of models trained with patterns extracted using the varying frame rate method is poorer (by about 1 per cent) than that of the models trained with patterns extracted using the linear compaction and elongation method. The multiple varying frame rates method gives a better performance (by about 11 per cent) than that of the linear compaction and elongation method. The multiple varying frame rates method improves the classification accuracy (by about 12 per cent) compared to the varying-frame rate method, since the duration of each phoneme in an utterance is taken into account to determine the value of frame shift for each phoneme in the utterance. The multiple varying frame rates method gives a comparable performance to that of the CDHMM-based classifiers. The multiple outerproduct matrices method performs better than the single outerproduct matrix method and the linear compaction and elongation method for mapping the varying length patterns to fixed length patterns. It is also seen that the augmented pattern extraction method gives a better performance than all other proposed methods.

Table 4. Classification accuracy of SVM models built using the fixed-length patterns based on the proposed approaches for E-set recognition task

Method for extraction of fixed-length patterns	Accuracy (per cent)
Linear compaction and elongation method	71.67
Single outerproduct matrix method	81.48
Multiple outerproduct matrices method	84.44
Augmented pattern extraction method	86.90
Varying frame shift method	70.56
Multiple varying frame shift method	82.63

The continuous density HMMs in the explicit kernel feature space give a better performance compared to the discrete HMMs in the kernel feature space is demonstrated. However, as the dimensionality of the kernel feature space representation is high, it is computationally intensive. Additionally, the continuous density HMMs cannot be constructed in the implicit kernel feature space as the method for density estimation in the kernel feature space using kernel functions on the data does not exist. However, it is interesting to note from studies in Section III and IV that the string kernel based SVMs for classification of sequence of codebook indices obtained using kernel-based clustering give a better performance than the continuous density HMMs in the kernel feature space.

5. CLASSIFICATION OF VARYING-LENGTH TEMPORAL PATTERNS REPRESENTED AS SETS OF FEATURE VECTORS

Modelling sets of vectors involve tasks such as speaker recognition, spoken language identification, audio classification, music classification and speech emotion

recognition[7,31,32]. In these tasks, the duration of examples is large and the local temporal dynamics is not critical. Production of different examples of the same class may have a different number of acoustic events. Classification of set of vectors can be done using parametric methods such as GMMs, or nonparametric methods such as nearest neighbour approach and vector quantisation-based approach. The nearest neighbour approach is used for speaker identification task[33]. In this work, the feature vectors of the registered speakers are stored as reference vectors. During testing, the feature vectors of the test utterance are compared with the feature vectors of each of the registered speakers. The speaker of the test utterance is the speaker of the feature vector set that gives the lowest distance. Vector quantisation (VQ)-based approach[34] is similar to nearest neighbour approach, except that the distance is measured to the nearest centroid that represents a cluster of feature vectors. Better approach is to model the feature vectors by a set of mean and covariance parameters. This technique is employed in models such as GMM and is commonly used in modelling sets of vectors[7]. The number of components of GMM is chosen empirically and the choice of optimal number of components is critical.

The motivation for the proposed approaches is to incorporate the advantages of generative models for the representation of the temporal data and the discriminatory approach for classification. The similarity-based paradigm is shown effective for classification tasks[35]. Two approaches in a hybrid framework, that first uses a generative model-based method to represent a varying-length sequence of feature vectors as a fixed-length pattern and then uses a discriminative model for classification, have been proposed. In the score vector-based approach, each temporal data in the training data set is modelled by a GMM. In tasks that involve modelling sets of vectors, the duration of the temporal data is large and hence a GMM can be built for each temporal data. The log-likelihood of a temporal data for a given GMM model is used as a score. A score vector is obtained by applying a temporal data to the GMM models of temporal data in the training set. A test temporal data is also represented using a score vector. An SVM-based classifier is then used for classification of the score vector representation of temporal data. In tasks that involve modelling, the set of vectors of a temporal data, though the local temporal dynamics is not critical, the sequence information present at the gross level needs to be modelled. In the score vector-based representation, the temporal dynamics in the temporal data is not modelled and the dimension of the fixed-length score vector depends on the cardinality of the training data set. To address these issues, a segment modelling-based approach is proposed. In this approach, a temporal data is segmented into a fixed number of segments. Each segment is modelled by a multivariate Gaussian model or a GMM. The model parameters of segments are concatenated in the order of the segments to form a parametric vector. Then, an SVM-based classifier is used to classify the parametric vector representation of temporal data. The proposed approaches are studied for the speech emotion recognition task.

5.1 Gaussian Mixture Model Score Vector-based Approach to Modelling Sets of Vectors

Gaussian mixture model is a linear superposition of Gaussian components. For a d-dimensional feature vector $\bar{x}$, the likelihood of $\bar{x}$ for a GMM with K components is defined as follows:

$$p(\bar{x} / \lambda) = \sum_{k=1}^{K} w_k p_k(\bar{x})$$

The component weights, w_k, satisfy the constraints, $\sum_{k=1}^{K} w_k = 1$ and $0 \le w_k \le 1$. Each of the K unimodal Gaussian densities, $p_k(\bar{x})$, is parameterised by a d-dimensional mean vector, $\bar{\mu}_k$, and a covariance matrix, C_k, as follows:

$$p_k(\bar{x}) = \frac{1}{(2\pi)^{d/2} |C_k|^{1/2}} e^{-\frac{1}{2}(\bar{x}-\bar{\mu}_k)^T C_k^{-1}(\bar{x}-\bar{\mu}_k)}$$

The parameters of the GMM are collectively denoted as follows:

$$\lambda = \{w_k, \mu_k, C_k\}, k = 1, 2, ..., K$$

The commonly used method for estimation of GMM parameters is the maximum likelihood (ML) method. The ML method suffers from the over-fitting if the model complexity is too high. Let a multivariate temporal data be denoted by a set of feature vectors $X_i = \{\bar{x}_{i1}, \bar{x}_{i2}, ..., \bar{x}_{ij}, ..., \bar{x}_{in_i}\}$, where X_i is a d-dimensional feature vector and n_i is the number of feature vectors. Let $D = \{X_1, X_2, ..., X_M\}$ be the training data set. For a temporal data, X_i, the likelihood score using a GMM with λ as the set of model parameters is defined as follows:

$$p(X_i / \lambda) = \prod_{n=1}^{n_i} p(\bar{x}_{in} / \lambda)$$

The log-likelihood is given by

$$\ln p(X_i / \lambda) = \sum_{n=1}^{n_i} \ln p(\bar{x}_{in} / \lambda)$$

In the proposed score vector-based approach, a GMM λ_i is built for each temporal data X_i in the training data set D. Then the log-likelihood score is computed for each of the M temporal data in D, by applying each of the models. A temporal data is now represented by an M-dimensional score vector that consists of M scores. Each score in a score vector is given by

$$S_{ij} = \ln p(X_i / \lambda_j) = \sum_{n=1}^{n_i} \ln \sum_{k=1}^{K} w_k p_k(\bar{x}_{in} / \lambda_j)$$

where $1 \le i, j \le M$

The steps of the proposed GMM score vector-based method for classification of temporal data are as follows:

- Build a GMM for each temporal data in the training data set.
- Apply each temporal data to all the M models

and form a score vector using the log likelihood scores.

- Normalise each score by dividing it with the length of the temporal data. An entry in the score vector for X_i is given by

$$S_{ij} = \frac{1}{n_i} \ln p(X_i / \lambda_j)$$

Score vector S_i for X_i is given by $S_i = [S_{i1}, S_{i2}, ..., S_{iM}]^T$. Such score vectors are the fixed-length patterns. Since the scores are log likelihood values, each score vector S_i is again normalised by $e^{(a*S_i)}$ where a is a constant that is chosen empirically.

- Build an SVM-based classifier using the GMM score vector-based representation of temporal data.
- Apply the test temporal data to all the M GMMs and get the score vector.
- Use the SVM-based classifier to classify the test temporal data represented using the GMM score vector.

The generation of GMM score vector is shown in Fig.7. The dimension of the fixed-length score vector depends on M, the cardinality of training data set. Methods to mitigate the problem of higher dimensionality in case of similarity-based classifiers have been suggested[35].

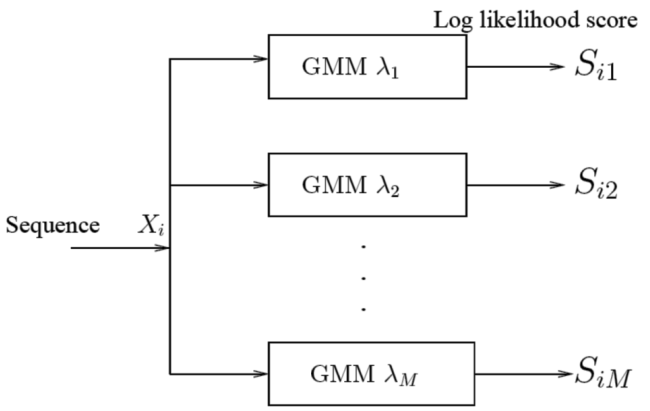

Figure 7. Score vector generation for a temporal data using GMMs.

5.2 Segment Modelling-based Approach to Modelling Sets of Vectors

In this approach, a temporal data is split into a fixed number of segments. Each segment is modelled by a multivariate Gaussian model or a Gaussian mixture model. The model parameters are concatenated in the order of the segments in a temporal data to form a fixed-dimensional parameter vector. The number of segments and length of each segment depends on the application. Figure 8 shows the sequence of steps in the segment modelling based approach.

Let a temporal data be denoted by $X_i = \{\bar{x}_{i1}, \bar{x}_{i2}, ..., \bar{x}_{ij}, ..., \bar{x}_{in_i}\}$, where $\bar{x}_{ij}$ is a d-dimensional feature vector and n_i is the number of feature vectors. Let a set of temporal data, $D = \{X_1, X_2, ..., X_M\}$, be the training data set. Let the fixed number of segments in each temporal data be L. For a temporal data, X_i, the fixed-dimensional parameter vector Z_i, of $L*(d+(d*(d+1)/2))$ dimension is formed by

$$Z = \{\bar{z}_1, \bar{z}_2, ..., \bar{z}_L\}$$

where $\bar{z}_l = [\bar{\mu}_l, C_l], l = 1, 2, ..., L$ and

$$\bar{\mu}_l = \frac{1}{N_l} \sum_{n=1}^{N_l} \bar{x}_n$$

$$C_l = \frac{1}{N_l} \sum_{n=1}^{N_l} (\bar{x}_n - \bar{\mu}_l)(\bar{x}_n - \bar{\mu}_l)^T$$

where N_l is the number of feature vectors in the l^{th} segment. Each $\bar{z}_l$ is of dimension $d+(d*(d+1)/2)$. For GMM based segment modelling, each segment is modelled using K mixtures. The parameters of the GMM for the l^{th} segment are

$$\lambda_l = \{w_{lk}, \bar{\mu}_{lk}, C_{lk}\}, k = 1, 2, ..., K$$

The Expectation-Maximisation (EM) algorithm was used to find the maximum likelihood estimates of the model parameters. In the case of GMM-based segment modelling, each $\bar{z}_l$ is of dimension K($d+(d*(d+1)/2)$). The parameter

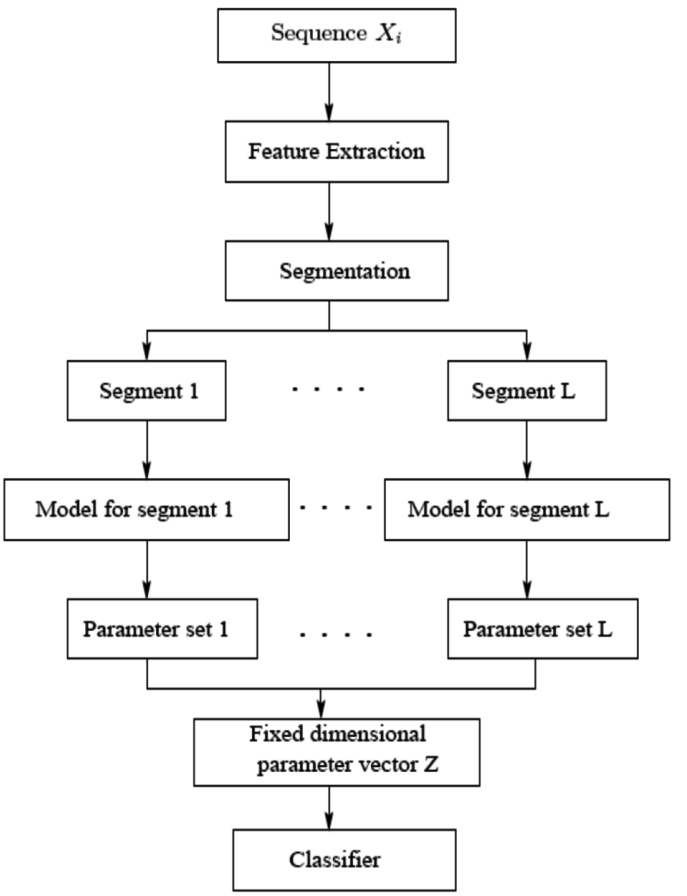

Figure 8. Block diagram of the segment modelling-based approach for varying-length temporal data classification.

vector is formed by concatenating the mean vectors of K mixtures and the upper triangular parts including the diagonal elements of the full covariance matrices of K mixtures corresponding to the l^{th} segment. Now, the temporal data, X_t, is represented as a fixed-dimensional parameter vector by concatenating all $\bar{z}_l$'s in the order of the segments in the temporal data. This helps to maintain the temporal ordering of the segments in a temporal data.

5.3 Speech Emotion Recognition

The proposed approaches were studied for the task of speech emotion recognition. The Berlin emotional speech database[36] was used in our studies. Five female and five male actors uttered ten sentences in German that had little emotional content textually. The database includes a total of 494 utterances from seven emotional classes: Fear(F), Disgust(D), Happiness(H), Boredom(B), Neutral(N), Sadness(S) and Anger(A). The duration of the utterances varied from one to two seconds. 80 per cent of the utterances were used for training and the remaining utterances were used for testing. Short-time spectral analysis was carried out to extract the sequence of feature vectors from an utterance. A frame size of 20 ms and a shift of 10 ms were used for feature extraction. The Mel frequency cepstral coefficient (MFCC) vector representing a given frame is a 39-dimensional vector, where the first 12 components are Mel frequency components and the 13th component is log energy. Remaining 26 components are delta and acceleration coefficients that capture the dynamics. The effectiveness of the MFCC features for speech emotion recognition is shown in[37]. A 39-dimensional MFCC feature vector was chosen to represent a frame in an utterance in all the studies. Table 5 shows the performance of the GMM score vector-based approach for different number of mixtures with full covariance matrices. Singularity problem could arise if the number of mixtures as 5 or above was assumed. The results show that the performance depends on the number of mixtures. The best performance was obtained for 3 mixtures.

Table 5. Classification accuracy using the GMM score vector-based approach for speech emotion recognition

No. of mixtures	Classification accuracy (per cent)
2	66.67
3	70.48
4	53.33

The performance of the segment modelling-based approach to speech emotion recognition is given in Table 6. A multivariate Gaussian model was used to model each segment in a temporal data. Experiments were carried out separately for both the full covariance matrix and diagonal covariance matrix parameters, and for different number of segments. The best performance was obtained for 5 segments per utterance with full covariance matrix parameters. Since the correlations among features were not modelled in the case of the temporal data representation that uses the diagonal covariance parameters, the performance was poor compared to that of the full covariance counterpart.

Table 6. Classification accuracy using the segment modelling-based approach with single Gaussian model for each segment for speech emotion recognition

No. of segments	Diagonal covariance		Full covariance	
	Dimension	Accuracy (per cent)	Dimension	Accuracy (per cent)
1	78	60.00	819	63.81
2	156	56.19	1638	66.67
3	234	60.00	2457	64.76
4	312	58.10	3276	71.43
5	390	58.10	4095	72.38
6	468	49.52	4914	65.71

A GMM with 2 mixtures and full covariance matrices was also used to model each segment. The performance is given in Table 7. Singularity problem arises if the number of mixtures was 5 or above. The performance is better compared to that of the representation that uses a single Gaussian model with the diagonal covariance parameters, but less compared to that of the representation that uses a single Gaussian model per segment with full covariance method. In Table 8, the performance of the score vector-based approach, the segment modelling-based approach and two other methods evaluated on the Berlin emotional speech data set is given.

Table 7. Classification accuracy for the segment modeling based approach using GMM with 2 mixtures per segment for speech emotion recognition

No. of segments	Dimension	Classification accuracy (per cent)
1	1638	53.33
2	3276	57.14
3	4914	54.29
4	6552	62.86

The first method is the GMM-based classifier with the maximum likelihood method used for parameter estimation. The GMM-based classifier gives the best performance for 25 mixtures. Choosing optimal number of mixtures is critical in the GMM-based classifier. In the second method, the variational Bayesian approach is used for parameter estimation in GMM (VBGMM). Using the variational Bayes framework to GMMs helps in training models with less amount of data and in determining the optimal number of components. It is free from the singularity problem that arises frequently in GMMs[38]. The number of mixture components of seven

Table 8. Classification accuracy of different approaches to speech emotion recognition

Classifier	Input to the classifier	Accuracy (per cent)
GMM	MFCC	64.73
VBGMM	MFCC	67.62
SVM	GMM score vector	70.48
SVM	1 Gaussian/segment (Diag Cov)	60.00
	1 Gaussian/segment (Full Cov)	72.38
	GMM with 2mixtures (Full Cov) /segment	62.86

emotion classes(Fear(F), Disgust(D), Happiness(H), Boredom(B), Neutral(N), Sadness(S) and Anger(A)) chosen by the VBGMM models were 13,12,14,16,12,10 and 14 respectively. The VBGMM-based classifier performs better than the GMM-based classifier.

In the proposed score vector-based approach, the effective discriminative ability of the similarity-based representation and the discriminative approach for classification helps in achieving a better performance than that of the GMM-based classifiers. In case of speech emotion data set, though the local temporal dynamics is not critical, some kind of sequence information is present at the gross level in the temporal data that needs to be modeled. In the proposed segment modelling-based approach that uses a single Gaussian with full covariance matrix parameters, the temporal dynamics of the segments in a temporal data is maintained to some extent and the correlations among the features within a segment are also modelled. Hence, this method performs better than the score vector-based approach and it outperforms all other methods used for comparison.

6. CONCLUSIONS

We have proposed approaches for classification of varying length temporal data that involves modelling sequences using DHMMs in the kernel feature space, string kernel-based SVMs and CDHMMs in the kernel feature space. Approaches for modelling sequences using fixed-length patterns derived from varying-length sequences of feature vectors, namely, linear compaction and elongation method, anchor point-based method, varying-frame shift method and outerproduct matrix-based method have also been proposed. The performance of proposed methods were studied for recognition of spoken utterances of letters in E-set. String kernel-based SVM classifier with the sequences of codebook indices obtained using clustering and vector quantisation in the Gaussian kernel feature space gives a better performance than all other proposed approaches. In case of approaches that involve modelling sets of vectors, a hybrid framework that uses a generative front-end for temporal data representation and a discriminative model for classification has been proposed. In this framework, two different approaches, namely, the GMM score vector-based approach and the segment modelling-based approach are proposed. These approaches are studied for the tasks of speech emotion recognition. Incorporating the temporal ordering of the segments in the temporal data in case of the segment modelling-based approach helps in achieving a better performance for the task of speech emotion recognition, compared to all other methods used for comparison. The segment modelling-based approach is suitable for temporal data classification tasks in which the temporal order of segments needs to be maintained.

REFERENCES

1. Lee, K.F. Automatic speech recognition: The development of the SPHINX system. Kluwer Academic Press, 1999.
2. Yacoubi, A.E.; Gilloux, M.; Sabourin, R. & Suen, C.Y. An HMM-based approach for off-line unconstrained handwritten word modeling and recognition. *IEEE Trans. Pattern Analy. Machine Intell.*, 1999, **21**(8), 752-60.
3. Rabiner, L.R. A tutorial on hidden Markov models and selected application in speech recognition. *Proceedings IEEE*, 1989, **77**(2), 257-86.
4. Ververidis, D. & Kotropoulos, C. Emotional speech classification using Gaussian mixture models and the sequential floating forward selection algorithm. *In* IEEE Int. Conference on Multimedia and Expo, July 2005. pp. 1500-503.
5. El Ayadi, M.H.; Kamel, M.S. & Karray, F. Speech emotion recognition using Gaussian mixture vector autoregressive models. *In* Proceeddings of IEEE Internationl Conference Acoustics, Speech and Signal Processing, Honolulu, Hawai, USA, April 2007. pp. 957-60.
6. Neiberg, D. & Elenius, K. Emotion recognition in spontaneous speech using GMMs. *In* Interspeech, Pittsburgh, Pennsylvania, Sept 2006.
7. Mishra, H. K. & Chandra Sekhar, C. Variational Gaussian mixture models for speech emotion recognition. *In* Internationl Conference on Advances in Pattern Recognition, Kolkata, India, February 2009. pp. 183-86.
8. Yu, D.; Deng, L.; He, X. & Acero, A. Large-margin minimum classification error training for large-scale speech recognition tasks. *In* Proceeddings IEEE International Conference Acoustics, Speech and Signal Processing, Honolulu, Hawai, USA, April 2007. pp. 1137-140.

9. Satish, D.S. & Chandra Sekhar, C. Discrete hidden Markov models in kernel feature space for speech recognition. *In* International Conference on Systemics, Cybernautics and Informatics, Hyderabad, India, February 2004. pp. 653–658.
10. Nwe, T.; Foo, S. & De Silva, L. Speech emotion recognition using hidden Markov models. *Speech Communication*, 2003, **41**(4), 603-23.
11. Vapnik, V. Statistical Learning Theory. New York, Wiley-Interscience, 1998.
12. Lafferty, J.; McCallum, A. & Pereira, F. Conditional random fields: Probabilistic models for segmenting and lebeling sequence data. *In* International Conference on Machine Learning, 2001. pp. 282-89.
13. McCallum, A.; Freitag, D. & Pereira, F. Maximum entropy Markov models for information extraction and segmentation. *In* International Conference on Machine Learning, 2000. pp. 591-98.
14. Gao, Y. & J. Kuo. Maximum entropy direct models for speech recognition. *IEEE Trans. Speech Audio Process.*, 2006, **14**(3), 873-81.
15. He, X.; Deng, L. & Chou, W. Discriminative learning in sequential pattern recognition. *IEEE Signal Process. Mag.*, 2008, **25**(5), 14-36.
16. Abou-Moustafa, K.T.; Cheriet, M. & Suen, C.Y. On the structure of hidden Markov models. *Pattern Recog. Lett.*, 2004, **25**, 829-73, .
17. Girolami, M. Mercer kernel based clustering in feature space. *IEEE Trans. Neural Networks*, 2002, **13**(3), 780-84 .
18. Taylor, J.S. & Cristianini, N. Kernel methods for pattern analysis. Cambridge, UK: Cambridge University Press, 2004.
19. Satish, D.S. Kernel-based clustering and vector quantisation for pattern classification. Indian Institute of Technology Madras, Chennai, India, 2005. (MS Thesis)
20. Tian, S.; Mu,S. & Yin, C. Sequence-similarity kernels for SVMs to detect anomalies in system calls. *J. Neurocomputing*, 2007, **70**, 859-66.
21. Lodhi, H.; Saunders, C.; Shawe-Taylor, C.; Cristianini, N. & Watkins, C. Text classification using string kernels *J. Mac. Learning Res.,* 2002, **2**, 419-44.
22. Leslie, C.; Eskin, E.; Weston, J. & Noble, W. S. Mismatch string kernels for SVM protein classification. *In* Neural Info. Process. System, 2003, pp. 1417-424.
23. ISOLET Corpus, "Release 1.1," *In* Centre for spoken language understanding, Oregon Graduate Institute, July 2000.
24. Ganapathiraju, A.; Hamaker, J. & Picone, J. Hybrid SVM/HMM architectures for speech recognition. *In* International Conference on Spoken Language Processing, Beijing, China, Nov. 2000, pp. 50407.
25. Lee, W.; Chandra Sekhar, C.; Takeda, K. & Itakura, F. Recognition of continuous speech segments of monophones using support vector machines. *In* Proceeding of International Conference Spoken Language Processing, Denver, Colorado, September 2002, pp. 2653-656.
26. Bazzi, I. & Katabi, D. Using support vector machines for spoken digit recognition. *In* Proceeddings International Conference Spoken Language Processing, October 2000. pp. 433-36.
27. Chandra Sekhar, C.; Takeda, K. & Itakura, F. Recognition of consonant-vowel (CV) units of speech in a broadcast news corpus using support vector machines. *In* International Workshop on Pattern Recognition using Support Vector Machines, Aug 2002. pp. 171-85.
28. Siva Rama Krishna Rao, J.Y.; Chandra Sekhar, C. & Yegnanarayana, B. Neural networks based approach for detection of vowel onset points. *In* Advances in Pattern Recognition and Digital Techniques, Dec 1999. pp. 316-20.
29. Chandra Sekhar, C. Neural network models for recognition of stop consonant-vowel (SCV) segments in continuous speech. Indian Institute of Technology Madras, India, 1996. (PhD Thesis)
30. Anitha, R.; Srikrishna Satish, D. & Chandra Sekhar, C. Outerproduct of trajectory matrix for acoustic modelling using support vector machines. *In* IEEE International Workshop on Machine Learning for Signal Processing, Sept 2004. pp. 355-63.
31. Guo, G. & Li, S.Z. Content based audio classification and retrieval by SVMs. *IEEE Trans. Neural Networks*, January 2003. **14**(1), 209-15.
32. Shami, M. & Verhelst, W. An evaluation of the robustness of existing supervised machine learning approaches to the classification of emotions in speech. *Speech Communication*, January 2007, **49**, 201-12.
33. Higgins, A.L.; Bahler, L.G. & Portar, J.E. Voice identification using nearest neighbour distance measure. *In* Proceeddings IEEE International Conference Acoustics, Speech and Signal Processing, Minneapolis, USA, Aug 1993. pp. 375-78.
34. Rosenberg, A.E. & Soong, F.K. Evaluation of a vector quantisation talker recognition system in a text independent and text dependent modes. *In* Proceeddings IEEE International Conference Acoustics, Speech and Signal Processing, Tokyo, Japan, April 1986. pp. 873-76.
35. Pekalska, E. & Duin, R. Dissimilarity representations allow for building good classifiers. *Pattern Recog. Lett.*, 2002, **23**, 943-56.
36. Burkhardt, F.; Paeschke, A. Rolfes, M. & Weiss, W.S.B. A database of German emotional speech. *In* Interspeech, Lisbon, Portugal, 2005, pp. 1517-520.
37. Sato, N. & Obuchi, Y. Emotion recognition using Mel-frequency cepstral coefficients. *J. Natural Lang. Process.*, 2007, **14**(4), 83-96, .
38. Bishop, C.M. Pattern recognition and machine learning. Springer, 2006.

Contributors

Dr C. Chandra Sekhar received his MTech (Elect Engg) and PhD (Comp Sci and Engg) from Indian Institute of Technology (IIT) Madras, Chennai in 1986 and 1997, respectively. Presently working as an Associate Professor, Dept of Comp Sci and Engg at IIT Madras, Chennai. His current research interests are in speech processing, handwritten character recognition, artificial neural networks, kernel methods, Bayesian methods and content-based information retrieval of multimedia data.

Ms S. Chandrakala received her BE (Electro and Comm Engg) from Madurai Kamaraj University in 1992. She received her MTech (Comp Sci and Engg) from SASTRA University, Thanjavur in 2002. Presently, she is pursuing PhD in the Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai. Her research interests include machine learning, time series analysis, kernel methods, speech, image and video processing.