International Journal of Speech Technology (2019) 22:231–249
https://doi.org/10.1007/s10772-018-09587-1

Segment-level probabilistic sequence kernel and segment-level pyramid match kernel based extreme learning machine for classification of varying length patterns of speech

Shikha Gupta[1] · Ahmed Karanath[1] · Kansul Mahrifa[1] · A. D. Dileep[1] · Veena Thenkanidiyoor[2]

Received: 10 August 2018 / Accepted: 22 December 2018 / Published online: 5 February 2019

Abstract
In this work, we address some issues in the classification of varying length patterns of speech represented as sets of continuous-valued feature vectors using kernel methods. Kernels designed for varying length patterns are called as dynamic kernels. We propose two dynamic kernels namely segment-level pyramid match kernel (SLPMK) and segment-level probabilistic sequence kernel (SLPSK) for classification of long duration speech, represented as varying length sets of feature vectors using extreme learning machine (ELM). SLPMK and SLPSK are designed by partitioning the speech signal into increasingly finer segments and matching the corresponding segments. SLPSK is built upon a set of Gaussian basis functions, where half of the basis functions contain class-specific information while the other half implicates the common characteristics of all the speech utterances of all classes. The computational complexity of SVM training algorithms is usually intensive, which is at least quadratic with respect to the number of training examples. It is difficult to deal with the immense amount of data using traditional SVMs. For reducing the training time of classifier we propose to use a simple algorithm namely ELM. ELM refers to a wider type of generalized single hidden layer feedforward networks (SLFNs) whose hidden layer need not be tuned. In our work, we proposed to explore kernel based ELM to exploit dynamic kernels. We study the performance of the ELM-based classifiers using the proposed SLPSK and SLPMK for speech emotion recognition and speaker identification tasks and compare with other kernels for varying length patterns. Experimental studies showed that proposed ELM-based approach offer a 10–12% of relative improvement over baseline approach, and a 3–9% relative improvement over ELMs/SVMs using other state-of-the-art dynamic kernels.

Keywords Varying length patterns · Extreme learning machine · Segment level probabilistic sequence kernel · Segment level pyramid match kernel · Speech emotion recognition · Speaker identification

1 Introduction

Short-time analysis of speech signal involves performing spectral analysis on each frame of about 20 ms duration and representing each frame by a real valued feature vector. The speech signal of an utterance with T frames is represented as a sequential pattern $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t$ is a feature vector for tth frame. The duration of the utterances varies from one utterance to another. Hence, the number of frames also differs from one utterance to another. In the tasks such as acoustic modeling of sub-word units of speech such as phonemes, triphones, and syllables, duration of the data is short and there is a need to model the temporal dynamics and correlations among the features in the sequence of feature vectors. The hidden Markov models (HMMs) (Rabiner and Juang 2003) are commonly used for sequential pattern

✉ Shikha Gupta
shikha_g@students.iitmandi.ac.in

Ahmed Karanath
ahmed_k@students.iitmandi.ac.in

Kansul Mahrifa
kansul_mahrifa_c@students.iitmandi.ac.in

A. D. Dileep
addileep@iitmandi.ac.in

Veena Thenkanidiyoor
veenat@nitgoa.ac.in

[1] School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, Kamand, H.P. 175001, India

[2] Department of Computer Science and Engineering, National Institute of Technology Goa, Ponda, Goa 403401, India

classification. On the other hand, in the tasks such as speaker identification, spoken language identification, and speech emotion recognition, the duration of the data is long and preserving sequence information is not critical. In such cases, a speech signal is represented as a set of feature vectors. The focus of this paper is on the classification of varying length patterns of long duration speech that are represented as sets of continuous valued feature vectors. Conventionally, Gaussian mixture models (GMMs) (Reynolds 1995) are used for classification of varying length patterns represented as sets of feature vectors. The maximum likelihood (ML) based method is commonly used for estimation of parameters of the GMM for each class. When the amount of the training data available per class is limited, robust estimates of model parameters can be obtained through maximum a posteriori adaptation of the class-independent GMM (CIGMM), which is also called as universal background model (UBM), to the training data of each class (Reynolds et al. 2000). The CIGMM or UBM is a large GMM trained using the training data of all the classes. An important issue with the GMM-based classifiers is that they are trained using non-discriminative learning based approaches. In this work, we propose to consider discriminative learning based classifiers that are expected to perform better for the varying length pattern classification task.

Neural network based classifiers and support vector machine (SVM) based classifiers are two important discriminative learning based classifiers. Traditionally, multilayer feed forward neural networks (MLFFNNs) are considered for building discriminative classifiers. However, In MLFFNNs manual tuning of various parameters and hyper parameters is needed which results in the slow training of network. To overcome these issues, recently extreme learning machine (ELM) (Huang et al. 2006, 2012) is proposed which is gaining immense popularity. The ELM is a quick and robust learning algorithm for single hidden layer feedforward networks (SLFNs) that exhibit good generalization. In ELM, the weights for the connections between the input and hidden layer neurons are initialized randomly. The weights in the output layer are analytically computed that reduces the training time significantly. The learning process in ELM comprises of two steps. The first step map the input to a high-dimensional space known as ELM space. In the second step, the high dimensional data is projected onto a space of class labels. The dimensionality of ELM space is mostly chosen empirically. Choosing the dimensionality of ELM space can be avoided by using kernel version of ELM (Alexandos et al. 2015). In kernel based ELM (KELM), the network hidden layer outputs are directly taken from the kernel matrix. This avoids the problem of random assignment of weight for the hidden layer. Another popular discriminative classifier is support vector machine (SVM) that is proven to exhibit good generalization. SVMs can also be seen as SLFN like KELM. One of the difference is that in KELM the number of hidden nodes is decided by the number of training examples, whereas, in SVMs the number of hidden nodes is decided by the number of support vectors that are resultant of optimization of SVM cost function (Huang 2014). Support vector machines are originally designed for two-class pattern classification. Multi-class pattern classification problems are commonly solved using a combination of two-class SVMs which are obtained using one-against-rest or one-against-one approach (Allwein et al. 2000). When the number of classes are large, number of SVM needs to build are also large. However, KELM implicitly handles multi-class classification. Due to its learning speed, high efficiency and ability to handle multi-class classification tasks we propose to consider dynamic kernel based ELM classifiers for speech emotion recognition and speaker identification tasks that deal with long duration varying length patterns of speech represented as the set of feature vectors.

Classification of varying length sets of feature vectors using KELM-based classifiers and SVM-based classifiers requires the design of a suitable kernel as a measure of similarity between a pair of sets of feature vectors. The kernels designed for varying length patterns are referred to as dynamic kernels (Dileep and Chandra Sekhar 2014). Fisher kernel using GMM-based likelihood score vectors (Smith et al. 2001), probabilistic sequence kernel (Lee et al. 2007), GMM supervector kernel (Campbell and Sturim 2006), GMM-UBM mean interval kernel (You et al. 2009), GMM-based intermediate matching kernel (Dileep and Chandra Sekhar 2014) and GMM-based pyramid match kernel (Dileep and Chandra Sekhar 2012) are some of the state-of-the-art dynamic kernels for sets of feature vectors. In this paper, we propose segment-level pyramid match kernel (SLPMK) (Gupta et al. 2016a) and segment-level probabilistic sequence kernel (SLPSK) (Gupta et al. 2016b) as dynamic kernels for speech signals represented as varying length sets of feature vectors.

In SLPMK speech signal is repeatedly divided into segments to form a pyramid of increasingly finer segments. Then the SLPMK between a pair of speech signals is constructed by matching the corresponding segments at every level of the pyramid. We explore two approaches to obtain SLPMK. The first approach is inspired by the spatial pyramid match kernel (Lazebnik et al. 2006) proposed for image classification. In this approach, each segment is represented as a bag-of-codewords, where the codewords are obtained by clustering all the feature vectors of all the speech signals using K-means clustering technique. The codebook-based SLPMK (CBSLPMK) between a pair of speech signals is computed as a weighted sum of the number of new matches found at different levels of the pyramid of segments. The bag-of-codewords representation used in CBSLPMK suffers from loss of information due to the hard assignment

of a feature vector to a codeword. To address this issue, we propose Gaussian mixture model (GMM) based SLPMK (GMMSLPMK) as the second approach to constructing the SLPMK. In this approach, bag-of-codewords representation for each segment of a speech signal is obtained by soft assignment of the feature vectors to codewords using class independent GMM as soft clustering technique. Further, we explore the probabilistic sequence kernel (PSK) (Lee et al. 2007) to include local information in matching the two speech utterances and to maintain the temporal ordering of the feature vectors. PSK maps a set of feature vectors onto a high dimensional probabilistic score space. The probabilistic score space for a class is obtained by using the posterior probability of components of adapted GMM built for that class and the posterior probability of component of class-independent Gaussian mixture model (CIGMM) to which the data of a class is adapted. PSK does not include temporal information while computing the kernel. In this work, we propose segment-level probabilistic sequence kernel (SLPSK) as the dynamic kernel for building the SVM-based classifier for classification of speech signals represented as varying length sets of feature vectors. We propose to divide each speech signal into the fixed number of segments. We propose to compute PSK of the local feature vectors of a particular segment from the two examples. Then the proposed SLPMK is computed as a combination of PSKs corresponding to all the segments. As the kernel is computed at the segment level, it is expected to include more local information. Salient features of the proposed SLPMK and SLPSK as compared to that of other state-of-the-art dynamic kernels for sets of feature vectors are: (i) maintaining the temporal ordering of the feature vectors in a speech signal for some extent, and (ii) using the local information for matching between two speech utterances represented as sets of feature vectors.

In this work, we propose to use dynamic kernel based ELM for speech emotion recognition and speaker identification tasks. The effectiveness of the proposed SLPSK and SLPMK is studied for speech emotion recognition and speaker identification tasks using KELM-based classifiers. The performance of KELM-based classifier using SLPMK and SLPSK is compared with that of the KELM-based classifier using other state-of-the-art dynamic kernels. The contribution of this work is as follows: (i) two dynamic kernels namely SLPMK and SLPSK are proposed that retain the ordering of feature vectors to some extent as well as use the local information for matching two speech utterances, (ii) KELM-based classifiers using dynamic kernel including the proposed SLPMK and SLPSK for speech emotion recognition and speaker identification tasks, and (iii) comparison with SVM-based classifiers using SLPMK and SLPSK for speech emotion recognition and speaker identification tasks.

The remainder of the paper is organized as follows. In Sect. 2 we present ELM for the classification of varying length patterns of speech represented as sets of feature vectors. In Sect. 3, a brief review of dynamic kernels for sets of feature vectors is presented. The proposed dynamic kernels i.e. SLPMK and SLPSK are discussed in Sects. 4 and 5 respectively. In Sect. 6, the studies on speech emotion recognition and speaker identification tasks are presented. The discussion of the proposed approach is presented in Sect. 7. The conclusions are presented in Sect. 8.

2 Extreme learning machine for varying length pattern of speech

In this section, we briefly explain the learning strategy of extreme learning machine (ELM) by Huang et al. (2006, 2012) and its extension to kernel ELM (KELM). ELM is a simple and efficient learning algorithm for single-hidden layer feedforward networks (SLFNs) in which input layer weights and bias are initialized randomly to obtain the output of the hidden layer which leads to fast network training and low human supervision. Additionally, the algorithm guarantees lowest training error and smallest norm of learned weights. After the input weights and the hidden layer biases are chosen arbitrarily, SLFNs can be simply considered as a linear system and the output weight matrix $\boldsymbol{\beta}$ (linking the hidden layer to the output layer) can be analytically determined through simple generalized inverse operation (Rao and Mitra 1971) of the hidden layer output matrices $\mathbf{H}$ as:

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{1}$$

where $\mathbf{T}$ is a matrix containing the expected network target label. The generalized inverse of a matrix can be calculated using orthogonal projection method, orthogonalization method, iterative method or singular value decomposition (SVD). The advantage of generalized inverse operation is that it avoids lengthy training phase where the parameters of the network are tuned iteratively with some appropriate learning parameter.

In the task like speaker identification and speech emotion recognition length of speech utterance varies from one example to other. In such cases input example $\mathbf{X}_i = \{\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{it}, \ldots, \mathbf{x}_{iT}\}$, is represented as varying length set of feature vector and training data is denoting as $\{\mathbf{X}_i, \mathbf{t}_i\}, i = 1, \ldots, L$ where L is the total number of training examples and $\mathbf{t}_i = [t_{i1}, \ldots, t_{ic}]^{\intercal}$ is the corresponding class label vector with $t_{ic} = 1$ if $\mathbf{X}_i$ belong to class c or $t_{ic} = -1$ otherwise. One way of training ELM network using the varying length data is by considering individual feature vector $\mathbf{x}_{it}$ as an example or by considering $\mathbf{x}_{it}$ as the super-vector of contextual vectors around $\mathbf{x}_{it}$. The second case is the standard in the speech community (Chen et al. 2015), where the input

feature vectors to the ELM network is obtained by stacking every d-dimensional feature vectors $\mathbf{x}_{it}$ by l contextual vectors to the left and r contextual vectors to the right. Thus, the total number of stacked frames is $l + r + 1$. Therefore, the dimension of input feature vectors to the ELM network is $d(l + r + 1)$ corresponding to every frame $\mathbf{x}_{it}$. Another approach is obtain a fixed-dimensional representation by mapping these set of feature vectors to bag-of-codeword representation. A varying length set of feature vector can be converted into bag-of codebook representation by using both soft and hard quantization techniques. Size of this representation is dependent on the size of the codebooks. This fixed length representation now can be used as input to the conventional ELM. This is an elegant approach. However, it leads to a loss of information due to quantization.

The issue with the naive implementation of ELM is that for very large training datasets, even though the entire algorithm is faster than other conventional methods (such as iterative tuning of weights as in multilayer feedforward neural networks), the inverse calculation requires lot of resources, since the entire training data has to be loaded onto memory. Also for avoiding the problem of time-consuming algorithms for the determination of ELM space dimensionality (number of hidden node), kernel versions of the ELM classifier have been recently proposed by Huang et al. (2012) and Alexandos et al. (2015). The idea of kernel ELM is that the network hidden layer outputs need not to be calculated by passing the training example as input, but they can be inherently encoded in the ELM kernel matrix defined by $\mathbf{K} = \mathbf{H}^{\intercal}\mathbf{H}$, where $\mathbf{H} \in \mathbb{R}^{L \times h}$ refers to the training data representations in the ELM space with L is the number of training example and h is the dimensionality of ELM space. The classification problem for ELM with multi-output nodes can be formulated as

$$\text{Minimize:} \quad \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|, \quad \|\boldsymbol{\beta}\| \tag{2}$$

where $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_j, \ldots \boldsymbol{\beta}_c]$ is the matrix of weights linking hidden layer to the output nodes. $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_i, \ldots, \mathbf{t}_L]$ is a matrix containing the expected network target vectors. $\mathbf{H}$ is the hidden-layer output matrix, $\mathbf{H} = [\mathbf{h}(\mathbf{X}_1), \ldots, \mathbf{h}(\mathbf{X}_i), \ldots, \mathbf{h}(\mathbf{X}_L)]$ where each $\mathbf{h}(\mathbf{X}_i)$ is the output vector of the hidden layer with respect to the input $\mathbf{X}_i$. $\mathbf{h}(\mathbf{X}_i)$ actually maps the data from the input space to the h-dimensional hidden-layer feature space (ELM feature space) $\mathbf{H}$, and thus, $\mathbf{h}(\mathbf{X})$ is indeed a feature mapping. Karush–Kuhn–Tucker (KKT) (Gordon and Tibshirani 2012) conditions are the first-order necessary conditions for a solution in nonlinear programming to be optimal, provided that some regularity conditions are satisfied. The least squares solution of Eq. (2) based on KKT condition can be written as:

$$\boldsymbol{\beta} = \mathbf{H}^{\intercal}\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^{\intercal}\right)^{-1}\mathbf{T} \tag{3}$$

where C is the regularization coefficient. For a test example $\mathbf{X}$, The output function of ELM for multi class classification is

$$f(\mathbf{X}) = \mathbf{h}(\mathbf{X})\mathbf{H}^{\intercal}\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^{\intercal}\right)^{-1}\mathbf{T} \tag{4}$$

If the feature mapping $\mathbf{h}(\mathbf{X})$ is unknown then the kernel matrix for ELM based on Mercers conditions can be defined as follows

$$\mathbf{K} = \mathbf{H}\mathbf{H}^{\intercal} \; : \; K(\mathbf{X}_i, \mathbf{X}_j) = \mathbf{h}(\mathbf{X_i})^{\intercal}\mathbf{h}(\mathbf{X_j}) \tag{5}$$

thus, for a test example $\mathbf{X}$, the output function $f(\mathbf{X})$of the kernel based extreme learning machine (KELM) can be written as

$$f(\mathbf{X}) = [K(\mathbf{X}, \mathbf{X}_1), \ldots, K(\mathbf{X}, \mathbf{X}_j)]\left(\frac{\mathbf{I}}{C} + \mathbf{K}\right)^{-1}\mathbf{T} \tag{6}$$

where $K(\mathbf{X}, \mathbf{X}_j)$ is the kernel function of hidden neurons of single hidden layer feedforward neural networks. In literature there are many kernel functions exist which satisfying the Mercer condition, such as linear kernel, polynomial kernel, Gaussian kernel, and exponential kernel. But these kernels can only be applied to fixed length patterns, and not on the varying length patterns. The kernels used for the varying length patterns are called as dynamic kernels (Dileep and Chandra Sekhar 2014). In this research work, we propose to explore dynamic kernel based ELM (DKELM) for simulation and performance analysis of the task such as speaker identification and speech emotion recognition. We explore the possibility of directly classifying varying length patterns with ELM using dynamic kernels, and compare its performance with state-of-the-art dynamic kernel based-SVM for speech emotion recognition and speaker identification tasks. Here the hidden layer feature mapping or the dimensionality of the hidden layer feature space (number of nodes in the hidden layer) need not be known, and a suitable dynamic kernel can be used. In the next section we present the brief review of state-of-the-art dynamic kernel for varying length pattern of speech.

3 Dynamic kernels for sets of feature vectors

In this section, we review the approaches to design dynamic kernels for varying length patterns represented as sets of feature vectors. Different approaches to design dynamic kernels are broadly divided into explicit mapping based approaches and matching based approaches (Dileep and Chandra Sekhar 2014).

3.1 Explicit mapping based approaches

These approaches involve mapping a set of feature vectors onto a fixed-dimensional representation and then defining a kernel function in the space of that representation. In this work we propose to explore Fisher kernel (FK) (Smith et al. 2001), GMM supervector (GMMSV) kernel (Campbell and Sturim 2006) and GMM-UBM mean interval (GUMI) kernel (You et al. 2009) as the dynamic kernels for sets of feature vectors constructed using the explicit mapping based approaches.

3.1.1 Fisher kernel

Fisher kernel (FK) (Smith et al. 2001) for sets of local feature vectors uses an explicit expansion into a kernel feature space defined by a GMM based likelihood score space in which a set of feature vectors is represented as a fixed dimensional Fisher score vector. Likelihood score space is formed using the first order derivatives of the log-likelihood with respect to the GMM parameters. For an utterance represented as a set of d-dimensional local feature vectors $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, the first order derivative of the log-likelihood, i.e., the gradient vector of the log-likelihood, with respect to mean vector of the qth component of the GMM, $\boldsymbol{\mu}_q$, is given by

$$\boldsymbol{\Psi}_q^{(\mu)}(\mathbf{X}) = \sum_{t=1}^{T} \gamma_q(\mathbf{x}_t)\mathbf{z}_{tq} \tag{7}$$

where $\mathbf{z}_{tq} = \Sigma_q^{-1}(\mathbf{x}_t - \boldsymbol{\mu}_q)$. Let the ith element of $\mathbf{z}_{tq}$ be denoted by z_{tiq}. Here, $\gamma_q(\mathbf{x}_t)$ is the responsibility of the component q for a local feature vector $\mathbf{x}_t$ and is given by

$$\gamma_q(\mathbf{x}_t) = \frac{w_q \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)}{\sum_{q'=1}^{Q} w_{q'} \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{q'}, \boldsymbol{\Sigma}_{q'})} \tag{8}$$

The gradient vector of the log-likelihood with respect to $\boldsymbol{\Sigma}_q$ is given by

$$\boldsymbol{\Psi}_q^{(\Sigma)}(\mathbf{X}) = \frac{1}{2}\sum_{t=1}^{T} \gamma_q(\mathbf{x}_t)\left[-\mathbf{u}_q + \mathbf{v}_{tq}\right] \tag{9}$$

where the d^2-dimensional vectors, $\mathbf{u}_q = \text{vec}\,(\boldsymbol{\Sigma}_q^{-1})$ and $\mathbf{v}_{tq} = \left[z_{t1q}\mathbf{z}_{tq}^\top, z_{t2q}\mathbf{z}_{tq}^\top, \ldots, z_{tdq}\mathbf{z}_{tq}^\top\right]^\top$. For any $d \times d$ matrix $\mathbf{A}$ with $a_{ij}, i, j = 1, 2, \ldots, d$ as its elements, $\text{vec}\,(\mathbf{A}) = [a_{11}, a_{12}, \ldots, a_{dd}]^\top$. The gradient of the log-likelihood with respect to w_q is given by

$$\boldsymbol{\Psi}_q^{(w)}(\mathbf{X}) = \sum_{t=1}^{T} \gamma_q(\mathbf{x}_t)\left[\frac{1}{w_q} - \frac{\gamma_1(\mathbf{x}_t)}{w_1\gamma_q(\mathbf{x}_t)}\right] \tag{10}$$

The Fisher score vector with respect to the parameters of the qth component of the GMM is obtained as a supervector of gradient vectors of the log-likelihood for that component and is given by

$$\hat{\boldsymbol{\Phi}}_q(\mathbf{X}) = \left[\boldsymbol{\Psi}_q^{(\mu)}(\mathbf{X})^\top, \boldsymbol{\Psi}_q^{(\Sigma)}(\mathbf{X})^\top, \boldsymbol{\Psi}_q^{(w)}(\mathbf{X})\right]^\top \tag{11}$$

Now, a set of local feature vectors $\mathbf{X}$ is represented as a fixed dimensional supervector $\boldsymbol{\Phi}_{\text{FK}}(\mathbf{X})$ of all the Q Fisher score vectors as follows:

$$\boldsymbol{\Phi}_{\text{FK}}(\mathbf{X}) = \left[\hat{\boldsymbol{\Phi}}_1(\mathbf{X})^\top, \hat{\boldsymbol{\Phi}}_2(\mathbf{X})^\top, \ldots, \hat{\boldsymbol{\Phi}}_Q(\mathbf{X})^\top\right]^\top \tag{12}$$

The dimension of Fisher score vector is $D = Q(d + d^2 + 1)$. The Fisher kernel between two sets of local feature vectors $\mathbf{X}_m$ and $\mathbf{X}_n$ is computed as

$$K_{\text{FK}}(\mathbf{X}_m, \mathbf{X}_n) = \boldsymbol{\Phi}_{\text{FK}}(\mathbf{X}_m)^\top \mathbf{F}^{-1} \boldsymbol{\Phi}_{\text{FK}}(\mathbf{X}_n) \tag{13}$$

Here $\mathbf{F}$ is the Fisher information matrix given as

$$\mathbf{F} = \frac{1}{L}\sum_{l=1}^{L} \boldsymbol{\Phi}_{\text{FK}}(\mathbf{X}_l)\boldsymbol{\Phi}_{\text{FK}}(\mathbf{X}_l)^\top \tag{14}$$

where L is the number of training samples. The computation of Fisher information matrix and its inverse is computationally intensive. For a 128-component GMM on the 39-dimensional feature vectors, the dimension of the resulting supervector of Fisher score vectors for an example is 19,98,081, and the dimension of the Fisher information matrix is $19{,}98{,}081 \times 19{,}98{,}081$.

3.1.2 GMM supervector kernel

The GMM supervector (GMMSV) kernel (Campbell and Sturim 2006) performs a mapping of a set of local feature vectors onto a higher dimensional vector corresponding to a GMM supervector. An example-specific adapted GMM is built for each example by adapting the means of the UBM using the data of that example. Let $\boldsymbol{\mu}_q^{(\mathbf{X})}$ be the mean vector of qth component in the example-specific adapted GMM for an example $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$. A GMM vector $\boldsymbol{\Psi}_q(\mathbf{X})$ for an example $\mathbf{X}$ corresponding to the qth component of GMM is obtained as follows:

$$\boldsymbol{\Psi}_q(\mathbf{X}) = \left[\sqrt{w_q}\boldsymbol{\Sigma}_q^{-\frac{1}{2}}\boldsymbol{\mu}_q^{(\mathbf{X})}\right]^\top \tag{15}$$

where w_q and $\boldsymbol{\Sigma}_q$ are the mixture coefficient and covariance matrix of qth component in UBM. The GMM supervector for the example $\mathbf{X}$ is given by

$$\boldsymbol{\Phi}_{\text{GMMSV}}(\mathbf{X}) = [\boldsymbol{\Psi}_1(\mathbf{X})^\top, \boldsymbol{\Psi}_2(\mathbf{X})^\top, \ldots, \boldsymbol{\Psi}_Q(\mathbf{X})^\top]^\top \tag{16}$$

The dimension of GMM supervector is $D = Qd$. The GMMSV kernel between a pair of examples $\mathbf{X}_m$ and $\mathbf{X}_n$ is given by

$$K_{\text{GMMSV}}(\mathbf{X}_m, \mathbf{X}_n) = \boldsymbol{\Phi}_{\text{GMMSV}}(\mathbf{X}_m)^\top \boldsymbol{\Phi}_{\text{GMMSV}}(\mathbf{X}_n) \tag{17}$$

3.1.3 GMM-UBM mean interval kernel

The GMM-UBM mean interval (GUMI) kernel (You et al. 2009) performs a mapping of a set of local feature vectors onto a higher dimensional vector corresponding to a GUMI supervector. In GUMI kernel, an example-specific adapted GMM is built for each example by adapting the mean vectors and covariance matrices of the UBM using the data of that example. Let $\boldsymbol{\mu}_q^{(\mathbf{X})}$ and $\boldsymbol{\Sigma}_q^{(\mathbf{X})}$ be the mean vector and the covariance matrix of qth component in the example-specific adapted GMM for an example $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$. A GUMI vector $\boldsymbol{\Psi}_q(\mathbf{X})$ for an example $\mathbf{X}$ corresponding to the qth component of GMM is obtained as follows:

$$\boldsymbol{\Psi}_q(\mathbf{X}) = \left(\frac{\boldsymbol{\Sigma}_q^{(\mathbf{X})} + \boldsymbol{\Sigma}_q}{2}\right)^{-\frac{1}{2}} \left(\boldsymbol{\mu}_q^{(\mathbf{X})} - \boldsymbol{\mu}_q\right) \tag{18}$$

where $\boldsymbol{\mu}_q$ and $\boldsymbol{\Sigma}_q$ are the mean vector and covariance matrix of qth component in UBM. The GUMI supervector is obtained by concatenating the GUMI vectors of different components as

$$\boldsymbol{\Phi}_{\text{GUMI}}(\mathbf{X}) = [\boldsymbol{\Psi}_1(\mathbf{X})^\top, \boldsymbol{\Psi}_2(\mathbf{X})^\top, \ldots, \boldsymbol{\Psi}_Q(\mathbf{X})^\top]^\top \tag{19}$$

The dimension of GUMI supervector is $D = Qd$. The GUMI kernel between a pair of examples $\mathbf{X}_m$ and $\mathbf{X}_n$ is given by

$$K_{\text{GUMI}}(\mathbf{X}_m, \mathbf{X}_n) = \boldsymbol{\Phi}_{\text{GUMI}}(\mathbf{X}_m)^\top \boldsymbol{\Phi}_{\text{GUMI}}(\mathbf{X}_n) \tag{20}$$

3.2 Matching based approaches

These approaches involve computing a kernel function by matching the feature vectors in the pair of sets of feature vectors. In this work, we propose to use CIGMM-based intermediate matching kernel (Dileep and Chandra Sekhar 2014), histogram intersection kernel (HIK) (Gemert et al. 2010) and Chi-square-χ^2 kernel (Vedaldi and Zisserman 2010) and GMM-based pyramid match kernel (Dileep and Chandra Sekhar 2012) as the dynamic kernels designed using the matching based approaches.

3.2.1 CIGMM-based intermediate matching kernel

An intermediate matching kernel (IMK) (Boughorbel et al. 2005) is constructed by matching the sets of feature vectors using a set of virtual feature vectors. For every virtual feature vector, a feature vector is selected from each set of feature vectors and a base kernel for the two selected feature vectors is computed. The IMK for a pair of sets of feature vectors is computed as a combination of these base kernels. In (Dileep and Chandra Sekhar 2014), the set of virtual feature vectors considered are in the form of the components of CIGMM. For every component of CIGMM, a feature vector each from the two sets of feature vectors, that has the highest probability of belonging to that component (i.e., value of responsibility term) is selected and a base kernel is computed between the selected feature vectors. The responsibility of qth component for a local feature vector $\mathbf{x}$, $\gamma_q(\mathbf{x})$, is given as

$$\gamma_q(\mathbf{x}) = \frac{w_q \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)}{\sum_{j=1}^{Q} w_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{21}$$

where w_q is the mixture coefficient of the component q, and $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$ is the normal density for the component q with mean vector $\boldsymbol{\mu}_q$ and covariance matrix $\boldsymbol{\Sigma}_q$. The local feature vectors $\mathbf{x}_{mq}^*$ and $\mathbf{x}_{nq}^*$ respectively in $\mathbf{X}_m$ and $\mathbf{X}_n$, are selected using the component q as

$$\mathbf{x}_{mq}^* = \arg\max_{\mathbf{x} \in \mathbf{X}_m} \gamma_q(\mathbf{x}) \quad \text{and} \quad \mathbf{x}_{nq}^* = \arg\max_{\mathbf{x} \in \mathbf{X}_n} \gamma_q(\mathbf{x}) \tag{22}$$

The CIGMM-based IMK is computed as the sum of the values of the base kernels computed for the Q pairs of selected local feature vectors as follows:

$$K_{\text{CIGMMIMK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{q=1}^{Q} k(\mathbf{x}_{mq}^*, \mathbf{x}_{nq}^*) \tag{23}$$

The Gaussian kernel $k(\mathbf{x}_{mq}^*, \mathbf{x}_{nq}^*) = \exp\left(-\delta ||\mathbf{x}_{mq}^* - \mathbf{x}_{nq}^*||^2\right)$ is used as the base kernel. Here δ is the width parameter of the Gaussian kernel that is empirically chosen.

3.2.2 Histogram intersection kernel

In histogram intersection kernel (HIK) (Gemert et al. 2010), a set of feature vectors is mapped onto a histogram vector. The histogram encoding of a set of feature vector is from soft quantization using CIGMM with Q components. Let $\mathbf{h}(\mathbf{X}_m)$ and $\mathbf{h}(\mathbf{X}_n)$ be the histogram vectors corresponding to the sets of feature vectors $\mathbf{X}_m$ and $\mathbf{X}_n$. The number of matches in the qth bin is given by histogram intersection function (Swain and Ballard 1991), defined as follows:

$$s_q = \min\left(h_q(\mathbf{X}_m), h_q(\mathbf{X}_n)\right) \tag{24}$$

An HIK is computed as the total number of matches and is given by,

$$K_{\text{HIK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{q=1}^{Q} s_q \tag{25}$$

3.2.3 GMM-based pyramid match kernel

In the pyramid match kernel (PMK), a set of feature vectors is mapped onto a multi-resolution histogram pyramid. The kernel is computed between a pair of examples by matching the pyramids using a weighted histogram intersection match function at each level of pyramid. In Dileep and Chandra Sekhar (2012), the CIGMMs built with increasingly larger number of components are used to construct the histograms at the different levels in the pyramid. At level j, a CIGMM of b^j components is built using the feature vectors in the training examples of all the classes. The histogram vectors $\mathbf{h}_j(\mathbf{X}_m)$ and $\mathbf{h}_j(\mathbf{X}_n)$ with b^j -dimensions, corresponding to the sets of feature vectors $\mathbf{X}_m$ and $\mathbf{X}_n$, is then obtained by soft quantization. An histogram intersection kernel, $K_{\text{HIK}}^{(j)}$ is then computed to obtain the number of matches between a pair of histogram vectors corresponding to a pair of examples $\mathbf{X}_m$ and $\mathbf{X}_n$ at each level, $j = 0, 1, \ldots, J-1$. Here, J is the total number of levels in the pyramid. The matching is a hierarchical process from the bottom of the pyramid to the top of the pyramid. The number of new matches at a level j is calculated by computing the difference between the number of matches at that level and the number of matches at its immediately higher level and is given by $K_{\text{HIK}}^{(j)}(\mathbf{X}_m, \mathbf{X}_n) - K_{\text{HIK}}^{(j+1)}(\mathbf{X}_m, \mathbf{X}_n)$. The number of new matches at each level is weighted according to the number of components of CIGMM at that level. The GMM-based PMK between a pair of examples is computed as a weighted sum of the number of new matches at different levels of pyramid and is given as,

$$K_{\text{PMK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{j=0}^{J-1} \frac{1}{b^{J-j}} (K_{\text{HIK}}^{(j)} - K_{\text{HIK}}^{(j+1)}) + K_{\text{HIK}}^{(J)} \tag{26}$$

In our studies, we compare the performance of the SVM-based classifiers using the proposed SLPMK and SLPSK with that of the SVM-based classifiers using kernels reviewed in this section.

4 Segment-level pyramid match kernels

In designing segment-level pyramid match kernels (SLPMKs), a speech utterance represented as a set of feature vectors is decomposed into pyramid of increasingly finer segments. SLPMK between a pair of speech utterances is computed by matching the corresponding segments at each level in the pyramid. Let $j = 0, 1, \ldots, J-1$ be the J levels in pyramid. At level 0 (i.e. $j = 0$) complete speech signal is considered as a segment. At level 1 (i.e. $j = 1$), a speech signal is divided into two equal segments. At level 2 (i.e. $j = 2$), a speech signal is divided into four equal segments and so on. Hence at any level j, a speech utterance is partitioned into 2^j equal segments.

4.1 Codebook-based SLPMK

For designing codebook based segment-level pyramid match kernel (CBSLPMK) we borrowed the idea from spatial pyramid match kernel (Lazebnik et al. 2006) which consider the pyramid of spatial division of images. In designing CBSLPMK, every segment from a speech utterance is mapped to a bag-of-codewords representation. A codeword is a representative feature vector for a group of similar feature vectors. Collection of all the codewords is known as a codebook. A codebook of size Q is constructed by clustering the feature vectors in the training examples of all the classes using K-means clustering technique. The bag-of-codewords representation for a speech segment is obtained by assigning every feature vector to one of the Q codewords. Let $\mathbf{h}_{jk}(\mathbf{X})$ be the Q-dimensional bag-of-codewords representation of kth segment of an example $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ in the jth level of pyramid. Let $h_{jkq}(\mathbf{X})$ be an element in the $\mathbf{h}_{jk}(\mathbf{X})$, indicating the number of feature vectors of k^{th} segment assigned to q^{th} codeword. Let $\mathbf{X}_m = \{\mathbf{x}_{m1}, \mathbf{x}_{m2}, \ldots, \mathbf{x}_{mT_m}\}$ and $\mathbf{X}_n = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, \ldots, \mathbf{x}_{nT_n}\}$ be the two sets of feature vectors. The number of matches in the qth codeword between the kth segments of $\mathbf{X}_m$ and $\mathbf{X}_n$ at jth level of pyramid is given by

$$s_{jkq} = \min\left(h_{jkq}(\mathbf{X}_m), h_{jkq}(\mathbf{X}_n)\right) \tag{27}$$

Total number of matches at level j between the kth segments of $\mathbf{X}_m$ and $\mathbf{X}_n$ is obtained as,

$$S_{jk} = \sum_{q=1}^{Q} s_{jkq} \tag{28}$$

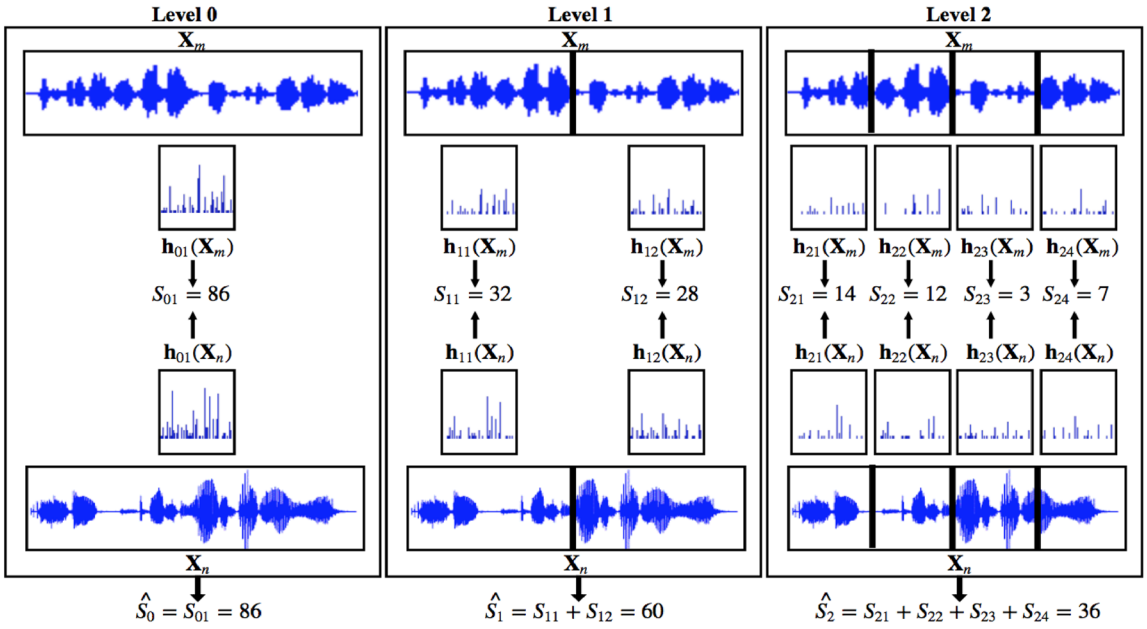
Total number of matches between $\mathbf{X}_m$ and $\mathbf{X}_n$ at level j is obtained as,

$$\hat{S}_j = \sum_{k=1}^{2^j} S_{jk} \tag{29}$$

Note that the number of matches found at level j also includes all the matches found at the finer level $j+1$. Therefore, the number of new matches found at level j is given by $\hat{S}_j - \hat{S}_{j+1}$. The CBSLPMK is computed as,

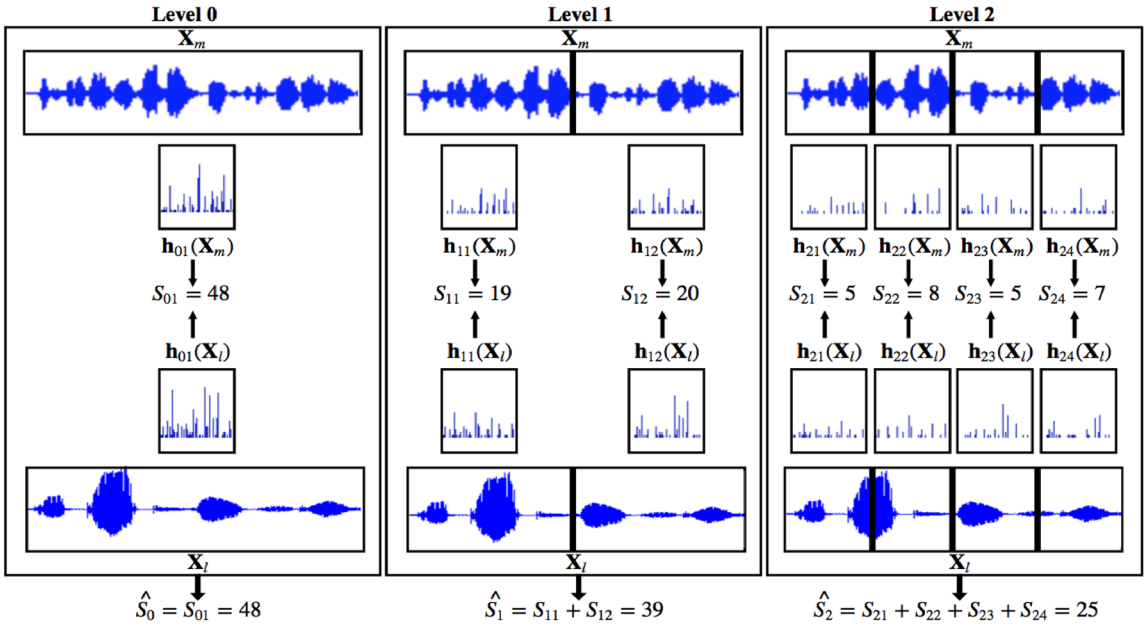
$$K_{\text{CBSLPMK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{j=0}^{J-2} \frac{1}{2^{J-(j+1)}} (\hat{S}_j - \hat{S}_{j+1}) + \hat{S}_{J-1} \tag{30}$$

Figure 1 illustrates the process of computing CBSLPMK between a pair of examples. Since kernel function is a similarity function which take two samples as input and spits



$$K_{\text{CBSLPMK}}(\mathbf{X}_m, \mathbf{X}_n) = \frac{1}{2^2}(\hat{S}_0 - \hat{S}_1) + \frac{1}{2^1}(\hat{S}_1 - \hat{S}_2) + \frac{1}{2^0}\hat{S}_2$$
$$= \frac{1}{4}(86 - 60) + \frac{1}{2}(60 - 36) + 36$$
$$= 54.50$$

(a) Segment level pyramids matching between two speech signal $\mathbf{X}_m$ and $\mathbf{X}_n$ of same class is performed using SLPMK.



$$K_{\text{CBSLPMK}}(\mathbf{X}_m, \mathbf{X}_l) = \frac{1}{2^2}(\hat{S}_0 - \hat{S}_1) + \frac{1}{2^1}(\hat{S}_1 - \hat{S}_2) + \frac{1}{2^0}\hat{S}_2$$
$$= \frac{1}{4}(48 - 39) + \frac{1}{2}(39 - 25) + 25$$
$$= 34.25$$

(b) Segmental pyramids matching between two speech signal $\mathbf{X}_m$ and $\mathbf{X}_l$ of different class is performed using SLPMK.

◂**Fig. 1** A schematic illustration of the construction of codebook-based SLPMK using segment-level pyramids that consists of three levels for a pair of examples. At level 0, a single segment of speech is considered resulting in a single bag-of-codewords representation. At level 1, a speech signal is subdivided into two segments, yielding two histograms, and so on

out how much similar are they. Same is observe from the Fig. 1, that when the two examples are from the same class, the value of CBSLPMK is higher than that for the examples from different classes. Examples of the same class are expected to be similar to each other rather then of other class.

The key issue in the design of SLPMK is the choice of the technique for constructing the bag-of-codewords representation for each segment of speech utterance. The *K*-means clustering method makes use of information about the centers of clusters and the distances of a feature vector to the centers of clusters to assign that feature vector to one of the clusters. A better bag-of-codewords representation of speech segment can be obtained by considering a clustering method that considers additional information like the spread of the clusters and the sizes of the clusters along with the centers of the clusters (Grauman and Darrell 2007). Moreover, the construction of CBSLPMK involves hard clustering. A better SLPMK is constructed by using soft clustering. In the next subsection, we propose the GMM-based SLPMK. The GMM uses the information about the spread and the size of the clusters along with the centers of the clusters for soft assignment of feature vectors.

4.2 GMM-based SLPMK

In this approach, we propose to use a class-independent GMM (CIGMM) for forming the clusters to obtain the bag-of-codewords representation for each speech segment. CIGMM is a large GMM of Q components built using the feature vectors in the training examples of all the classes. Every component of the CIGMM represents a codeword. The qth codeword is now represented by the mean vector $\boldsymbol{\mu}_q$, covariance matrix $\boldsymbol{\Sigma}_q$ and mixture weight ω_q of the qth component of CIGMM. The soft assignment of a feature vector from a segment to the qth component in the CIGMM is obtained using the responsibility term and it is given by

$$\gamma_q(\mathbf{x}_t) = \frac{w_q \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)}{\sum_{q'=1}^{Q} w_{q'} \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{q'}, \boldsymbol{\Sigma}_{q'})} \tag{31}$$

where $\mathcal{N}(\mathbf{x}_t \mid \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$ is the normal density for the component q. For the kth speech segment at jth level of pyramid, the effective number of feature vectors $h_{jkq}(\mathbf{X})$ assigned to a component q is given by

$$h_{jkq}(\mathbf{X}) = \sum_{t=1}^{T_k} \boldsymbol{\gamma}_q(\mathbf{x}_t) \tag{32}$$

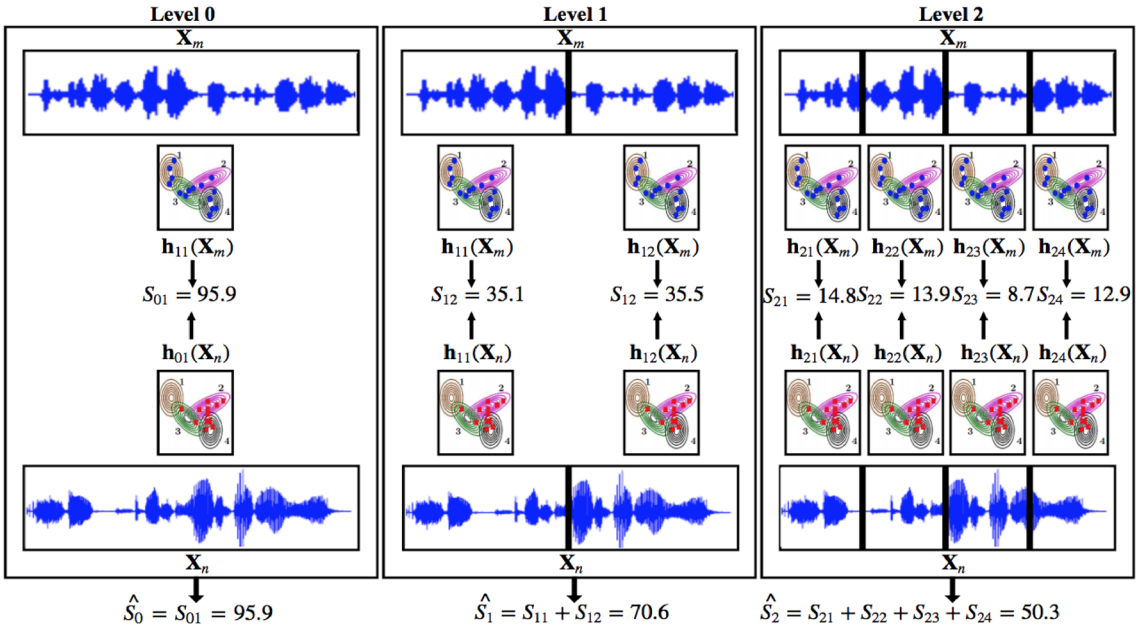
where T_k is the number of feature vectors in the kth segment of $\mathbf{X}$. For a pair of examples represented as sets of feature vectors, $\mathbf{X}_m$ and $\mathbf{X}_n$, number of matches in the qth codeword between the kth segments of $\mathbf{X}_m$ and $\mathbf{X}_n$ at jth level of pyramid is denoted by s_{jkq}, total number of matches at level j between the kth segments S_{jk} and total number of matches between $\mathbf{X}_m$ and $\mathbf{X}_n$ at level j, $\hat{S}_j$ are computed as in (27), (28) and (29) respectively. The GMM-based SLPMK (GMMSLPMK) between a pair of examples $\mathbf{X}_m$ and $\mathbf{X}_n$, K_{GMMSLPMK} is then computed as in eq. (30).

Figure 2 illustrates the process of computing GMMSLPMK between a pair of examples. In Fig. 2a, $\mathbf{X}_m$ and $\mathbf{X}_n$ are the two examples of same class are considered whereas in Fig. 2b, $\mathbf{X}_l$ is of different class is considered. It is observed from Fig. 2a, when the two examples are from the same class the value of GMMSLPMK matching score is higher in compare to the examples from different classes Fig. 2b. As the process of obtaining bag-of-codeword representation is different in CBSLPMK and GMMSLPMK, it is also seen from the Fig. 1 and Fig. 2 that the kernel values (the matching scores) are also different. For the same examples ($\mathbf{X}_m$–$\mathbf{X}_n$) and ($\mathbf{X}_m$–$\mathbf{X}_l$) matching scores from CBSLPMK and GMMSLPMK is computed in Figs. 1 and 2. It is observed in Fig. 2 that GMMSLPMK is more efficient as in GMMSLPMK matching score of examples belonging to same class is higher and examples of different classes is lower in compare to matching scores of CBSLPMK Fig. 1.

Both CBSLPMK and GMMSLPMK are valid positive definite kernel. The main advantages of using SLPMK over other dynamic kernels, especially over GMMPMK (Dileep and Chandra Sekhar 2012) are: (i) use of local information while matching a pair of speech utterances and (ii) maintaining temporal ordering of feature vectors in a speech utterance for some extent by matching at segment levels.

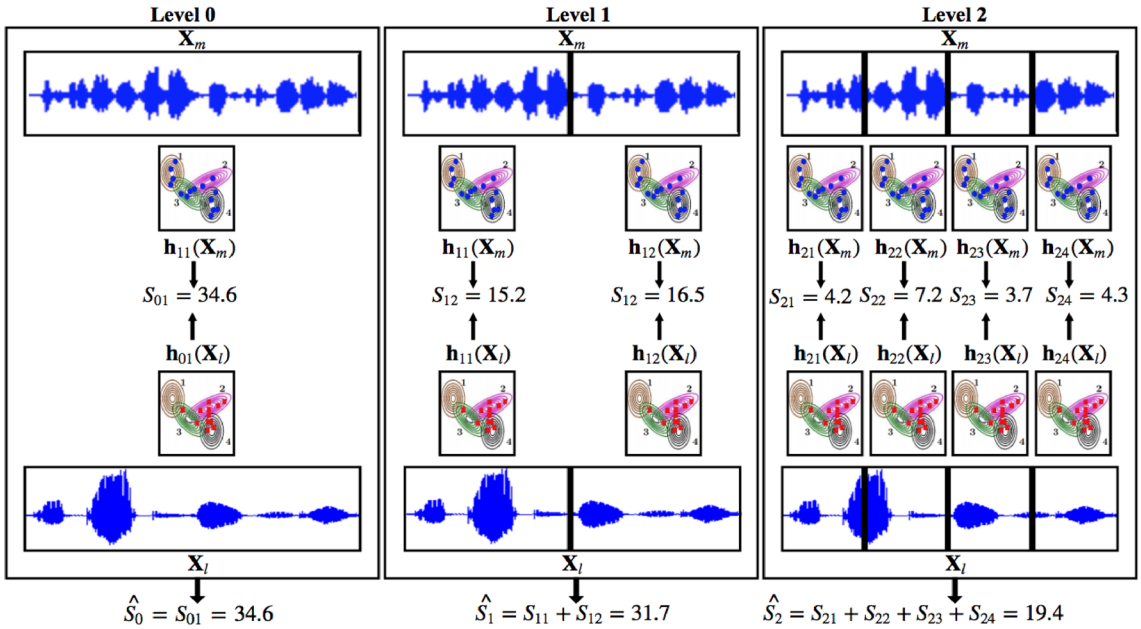
5 Probabilistic sequence kernel for sets of feature vectors

In this section we present probabilistic sequence kernel (PSK) constructed between pair of examples represented as sets of feature vectors. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ be a set of local feature vectors. PSK (Lee et al. 2007) maps a sets of feature vectors onto a fixed dimensional probabilistic feature vector obtained using Gaussian mixture model (GMM). The PSK uses universal background model (UBM) with Q components and the class-specific GMMs obtained by adapting the UBM. The UBM, also called as class independent



$$K_{\text{GMMSLPMK}}(\mathbf{X}_m, \mathbf{X}_n) = \frac{1}{2^2}(\hat{S}_0 - \hat{S}_1) + \frac{1}{2^1}(\hat{S}_1 - \hat{S}_2) + \frac{1}{2^0}\hat{S}_2$$
$$= \frac{1}{4}(95.9 - 70.6) + \frac{1}{2}(70.6 - 50.3) + 50.3$$
$$= 66.78$$

(a) Segment level pyramids matching between two speech signal $\mathbf{X}_m$ and $\mathbf{X}_n$ of same class is performed using SLPMK.



$$K_{\text{GMMSLPMK}}(\mathbf{X}_m, \mathbf{X}_l) = \frac{1}{2^2}(\hat{S}_0 - \hat{S}_1) + \frac{1}{2^1}(\hat{S}_1 - \hat{S}_2) + \frac{1}{2^0}\hat{S}_2$$
$$= \frac{1}{4}(34.6 - 31.7) + \frac{1}{2}(31.7 - 19.4) + 19.4$$
$$= 26.27$$

(b) Segmental pyramids matching between two speech signal $\mathbf{X}_m$ and $\mathbf{X}_l$ of different class is performed using SLPMK.

◂**Fig. 2** A schematic illustration of the construction of GMM-based SLPMK using segment-level pyramids that consists of 3 levels for a pair of examples. At level 0, a single segment of speech is considered resulting in a single bag-of-codewords representation. At level 1, a speech signal is subdivided into two segments, yielding two histograms, and so on

GMM (CIGMM), is a large GMM built using the training data of all the classes. A local feature vector $\mathbf{x}$ is represented in a higher dimensional feature space as a vector of responsibility terms of the $2Q$ components (Q from a class-specific adapted GMM and other Q from UBM), $\boldsymbol{\Psi}(\mathbf{x}) = [\gamma_1(\mathbf{x}), \gamma_2(\mathbf{x}), \ldots, \gamma_{2Q}(\mathbf{x})]^\top$. Since the element $\gamma_q(\mathbf{x})$ indicates the probabilistic alignment of $\mathbf{x}$ to the qth component, $\boldsymbol{\Psi}(\mathbf{x})$ is called the probabilistic alignment vector. Thus a probabilistic alignment vector includes the information specific to a class as well as the global information common to all the classes. A set of local feature vectors $\mathbf{X}$ is represented as a fixed dimensional vector $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X})$, and is given by

$$\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}) = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{\Psi}(\mathbf{x}_t) \tag{33}$$

Then, the PSK between two examples $\mathbf{X}_m = \{\mathbf{x}_{m1}, \mathbf{x}_{m2}, \ldots, \mathbf{x}_{mT_m}\}$ and $\mathbf{X}_n = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, \ldots, \mathbf{x}_{nT_n}\}$ is given as

$$K_{\text{PSK}}(\mathbf{X}_m, \mathbf{X}_n) = \boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}_m)^\top \mathbf{S}^{-1}\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}_n) \tag{34}$$

where $\mathbf{S}$ is the correlation matrix. The PSK in Lee et al. (2007), does not include temporal ordering of the local feature vectors. In many speech application, including the temporal information helps to build a better classifier. Also in many applications, preserving local information also helps to build a better discriminative classifier (Sachdev et al. 2015). In the following section, we propose segment-level PSK (SLPSK) to include local information as well as temporal information in the computation of PSK. It is seen from (33) that $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X})$ is obtained by pooling all the probabilistic alignment vectors corresponding to each local feature vectors of $\mathbf{X}$ and taking their average. This is called average pooling (Wang et al. 2010). In the next section we also propose to explore different pooling technique such as sum pooling (Wang et al. 2010) and max pooling (Yang et al. 2009) in the construction of $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X})$.

5.1 Segment-level probabilistic sequence kernel

In this section, we propose segment-level PSK (SLPSK). In SLPSK, speech utterance represented as a set of feature vectors is divided into a fixed number of segments and then feature vectors of each segment is mapped onto probabilistic feature vector. SLPSK between a pair of speech utterances is computed by matching the corresponding segments.

Let $\mathbf{X}_m = \{\mathbf{x}_{m1}, \mathbf{x}_{m2}, \ldots, \mathbf{x}_{mT_m}\}$ and $\mathbf{X}_n = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, \ldots, \mathbf{x}_{nT_n}\}$ be the sets of feature vectors for two examples (speech utterances). Let N be the number of segments into which each utterance is divided. Let $\mathbf{X}_m^k = \{\mathbf{x}_{m1}^k, \mathbf{x}_{m2}^k, \ldots, \mathbf{x}_{mT_m^k}^k\}$ and $\mathbf{X}_n^k = \{\mathbf{x}_{n1}^k, \mathbf{x}_{n2}^k, \ldots, \mathbf{x}_{nT_n^k}^k\}$ be the subsets of local feature vectors of $\mathbf{X}_m$ and $\mathbf{X}_n$ belonging to kth segment in their respective speech utterance. We propose to compute PSK between the two subsets of local feature vectors in the kth segment. The corresponding fixed dimensional vectors $\boldsymbol{\Phi}_{\text{PSK}}^k(\mathbf{X}_m^k)$ and $\boldsymbol{\Phi}_{\text{PSK}}^k(\mathbf{X}_n^k)$ are obtained by pooling their respective probabilistic alignment vectors. The different pooling techniques are presented in the end of this section. The segment-specific PSK between $\mathbf{X}_m^k$ and $\mathbf{X}_n^k$ is computed using

$$K_{\text{PSK}}^k(\mathbf{X}_m^k, \mathbf{X}_n^k) = \boldsymbol{\Phi}_{\text{PSK}}^k(\mathbf{X}_m^k)^\top \mathbf{S}_k^{-1}\boldsymbol{\Phi}_{\text{PSK}}^k(\mathbf{X}_n^k) \tag{35}$$

The correlation matrix $\mathbf{S}_k$ is defined as follows

$$\mathbf{S}_k = \frac{1}{M_k}\mathbf{R}_k^\top \mathbf{R}_k \tag{36}$$

where $\mathbf{R}_k$ is the matrix whose rows are the probabilistic alignment vectors for local feature vectors of kth segment and M_k is the total number of local feature vectors in kth segment. The SLPSK for the $\mathbf{X}_m$ and $\mathbf{X}_n$ is then computed as combination of the segment-specific PSKs as follows:

$$K_{\text{SLPSK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{k=1}^{N} K_{\text{PSK}}^k(\mathbf{X}_m^k, \mathbf{X}_n^k) \tag{37}$$

Since, PSK is a valid positive semidefinite kernel (Lee et al. 2007), the segment specific PSK is also a valid positive semidefinite kernel. Hence, the SLPSK is also a valid positive semidefinite kernel because the sum of valid positive semidefinite kernel is a valid positive semidefinite kernel.

Next, we discuss different pooling techniques used for pooling the probabilistic alignment vectors of sets of local feature vectors corresponding to each segments.

5.1.1 Pooling techniques for constructing $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X})$

Let $\mathbf{X}^k = \{\mathbf{x}_1^k, \mathbf{x}_2^k, \ldots, \mathbf{x}_{T^k}^k\}$ be the segment-level feature vectors corresponding the k^{th} segment of an utterance. In this work we propose to explore 3 pooling techniques that are popular in the image domain (Yang et al. 2009). They are:

(i) Average pooling: In average pooling, $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k)$ is obtained by pooling all the probabilistic alignment

vectors corresponding to each local feature vectors of $\mathbf{X}^k$ and taking their average. It is given as

$$\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k) = \frac{1}{T^k}\sum_{t=1}^{T^k}\boldsymbol{\psi}(\mathbf{x}_t^k) \tag{38}$$

(ii) Sum pooling: In sum pooling, $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k)$ is obtained by adding all the probabilistic alignment vectors corresponding to each local feature vectors of $\mathbf{X}^k$. It is given by:

$$\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k) = \sum_{t=1}^{T^k}\boldsymbol{\psi}(\mathbf{x}_t^k) \tag{39}$$

The $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k) = [\boldsymbol{\Phi}_1(\mathbf{X}^k), \boldsymbol{\Phi}_2(\mathbf{X}^k), \ldots \boldsymbol{\Phi}_{2\text{Q}}(\mathbf{X}^k)]^T$ is then normalized using sum normalization as suggested in Wang et al. (2010). The normalized qth value of $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k)$ is given as:

$$\boldsymbol{\Phi}_q(\mathbf{X}^k) = \frac{\boldsymbol{\Phi}_q(\mathbf{X}^k)}{\sum_{j=1}^{2Q}\boldsymbol{\Phi}_q(\mathbf{X}^k)} \tag{40}$$

(iii) Max pooling: In max pooling, $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k)$ is obtained by taking maximum of each dimension of all probabilistic alignment vectors corresponding to each local feature vectors of $\mathbf{X}^k$. It is given by

$$\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k) = \max(\boldsymbol{\psi}(\mathbf{x}_1^k), \boldsymbol{\psi}(\mathbf{x}_2^k), \ldots, \boldsymbol{\psi}(\mathbf{x}_t^k), \ldots, \boldsymbol{\psi}(\mathbf{x}_{T^k}^k)) \tag{41}$$

The $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k) = [\boldsymbol{\Phi}_1(\mathbf{X}^k), \boldsymbol{\Phi}_2(\mathbf{X}^k), \ldots \boldsymbol{\Phi}_{2\text{Q}}(\mathbf{X}^k)]^T$ is then normalized using l_2 normalization as suggested in Yang et al. (2009). The normalized qth value of $\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k)$ is given as:

$$\boldsymbol{\Phi}_q(\mathbf{X}^k) = \frac{\boldsymbol{\Phi}_q(\mathbf{X}^k)}{||\boldsymbol{\Phi}_{\text{PSK}}(\mathbf{X}^k)||_2} \tag{42}$$

In the next section, we present the effectiveness of the proposed kernels for speech emotion recognition and speaker identification tasks using ELM-based classifiers.

6 Experimental studies on speech emotion recognition and speaker identification

Speech emotion recognition task involves automatically identifying the emotional state of a speaker from his/her voice. Speaker identification task involves identifying a speaker among a known set of speakers using a speech utterance produced by the speaker. We first discuss the features and datasets used for the studies on speech emotion recognition and speaker identification. We have considered Mel frequency cepstral coefficients (MFCC) as features. The MFCC are the most successful and extensively used features for speech recognition. A speech utterance is represented by a set of feature vectors by extracting 39-dimensional feature vectors from every frame by performing spectral analysis. Among the 39 features, the first 12 features are the MFCC and the 13th feature is the log energy. The remaining 26 features are the delta and acceleration coefficients. A frame size of 20 ms and a shift of 10 ms are used for feature extraction from the speech signal of an utterance. The Berlin emotional speech database (Emo-DB) (Burkhardt et al. 2005) and the German FAU Aibo emotion corpus (FAU-AEC) (Steidl 2009) are used for studies on speech emotion recognition task. Emo-DB contains 494 utterances belonging to the following seven emotional categories with the number of utterances for the category given in parentheses: fear (55), disgust (38), happiness (64), boredom (79), neutral (78), sadness (53), and anger (127). These utterances correspond to ten sentences in German language uttered by five male and five female actors. We have considered 80% of the utterances for training and the remaining for testing. The multi-speaker speech emotion recognition accuracy presented in this work for the Emo-DB is the average classification accuracy along with 95% confidence interval (CI) obtained for 5-fold stratified cross-validation. We have considered four super classes of emotions, anger, emphatic, neutral, and motherese in the FAU-AEC. We have considered an almost balanced subset of the corpus defined for these four classes by CEICES of the Network of Excellence HUMAINE funded by the European Union (Steidl 2009). We perform the classification at the chunk (speech utterance) level in the Aibo chunk set. The speaker-independent speech emotion recognition accuracy presented in this study for the FAU-AEC is the average classification accuracy along with 95% CI obtained for 3-fold stratified cross validation. The 3-fold cross validation is based on the three splits defined in Appendix A.2.10 of Steidl (2009).

The studies on the speaker identification are performed on the 2002 and 2003 NIST speaker recognition (SRE) corpora (NIS 2002, 2003). We considered the 122 male speakers that are common to the 2002 and 2003 NIST SRE corpora. Each utterance in the training and test sets is divided into segments of around 5 s. Each speech segment is considered as an example. This leads to a total of 6661 examples with each speaker class having about 55 examples. The experiments are conducted in five trials by considering randomly chosen 30 utterances from each speaker class (total of 3660 examples) for training and rest for testing (total of 3001 examples). The speaker identification accuracy presented is the average classification accuracy along with 95% CI obtained for five trials.

The classification accuracy gives the percentage of test examples that are correctly predicted by the classifier. The classification accuracy is given as the ratio of number of test examples correctly classified ($\mathcal{L}_{correct}$) to the total number of test examples ($\mathcal{L}_{test}$). The classification accuracy (CA) in % is given as:

$$\text{CA (in \%)} = \frac{\mathcal{L}_{correct}}{\mathcal{L}_{test}} \times 100 \tag{43}$$

In order to ascertain the statistical importance of the result, the classification accuracy is presented along with the 95% CI. A simple asymptotic method (Wald method) Newcombe (1998) is employed to estimate the 95% CI of the classification accuracy. The CI of classification accuracy is computed as

$$\text{CI} = z\sqrt{\frac{\alpha(1-\alpha)}{\mathcal{L}_{test}}} \tag{44}$$

where α is the accuracy in decimals, and $\mathcal{L}_{correct}$ is the number of test examples. Here z is the standard normal distribution associated with a two-tailed probability. For 95% CI, z takes the value of 1.96.

We first present the experimental studies on speech emotion recognition and speaker identification using conventional ELM in Sect. 6.1. In Sect. 6.2 experimental studies using ELM-based classifiers with the proposed dynamic kernels, SLPMKs and SLPSKs are presented. In Sect. 6.3, we compare the results of SVM based classifier using proposed kernels with that of the proposed dynamic kernel based ELM classifiers. Comparison of results with state-of-the-art-approaches is presented in Sect. 6.4.

6.1 Experimental studies using conventional ELM

We consider the same architecture as discussed in Sect. 2 for the conventional ELM to perform classification of varying length patterns of speech. Every speech frame is represented as 39-dimensional MFCC vectors. As per the standards (Chen et al. 2015), we have considered $l = 7$ contextual vectors (frames) to the left and $r = 7$ contextual vectors (frames) to the right. Thus, the total number of stacked frames is 15. Now, the dimension of input feature vector to the conventional ELM is $D = 585$ corresponding to every frame. Thus there are 585 nodes in the input layer of the conventional ELM. Experiments are carried out using different number of nodes (h) in the hidden layer. A sigmoid activation function is considered for the nodes in the hidden layer. Table 1 shows the classification accuracy (in %) for speech emotion recognition and speaker identification tasks using the conventional ELM. It is observed that in all cases $h = 2048$ has given highest accuracy.

Table 1 Classification accuracy (CA) (in %) of the conventional ELM-based classifier for speech emotion recognition (SER) and speaker identification (Spk-id) tasks

SER				Spk-id	
EmoDB		FAU-AEC			
h	CA 95% CI	h	CA 95% CI	h	CA 95% CI
512	52.60 ± 0.23	512	44.25 ± 0.16	512	50 ± 0.18
1024	57.85 ± 0.21	1024	46.78 ± 0.14	1024	54 ± 0.19
2048	$\mathbf{59.25 \pm 0.29}$	2048	$\mathbf{48.60 \pm 0.17}$	2048	$\mathbf{59 \pm 0.16}$

Here, CA 95% CI indicates average classification accuracy along with 95% CI and h is the number of hidden nodes in ELM. Bold numerals indicate the best accuracy with respect to corresponding dataset and task

In another experiment, varying length speech utterance is mapped to a bag-of-codewords representation (description for bag-of-codewords representation is given in Sects. 4.1 and 4.2). Table 2 presents classification accuracy (CA) (in %) using conventional ELM considering bag-of-codeword representations of speech signals as input for speech emotion recognition (SER) and speaker identification (Spk-id) tasks. We consider K-means and GMM-based clustering techniques to obtain codebooks. Experiments are carried out using different number of codewords (Q) in a codebook. The dimension of input feature vector to the conventional ELM is $D = Q$ corresponding to every speech sample. Thus there are D nodes in the input layer of the conventional ELM. Experiments are also carried out using different number of nodes (h) in the hidden layer. Here, sigmoid activation function is considered for the nodes in the hidden layer. The accuracies presented in Table 2 are the best accuracies observed for the different values of (Q, h). The best performances in all these cases for the different tasks are shown using bold phase. It is observed that conventional ELM considering GMM-based bag-of-codeword representations for speech signals performed significantly better than that of the K-means based bag-of-codeword representations. It is also observed that conventional ELM performed better when the speech signals are presented using bag-of-codeword representations than that of the contextual vector representation.

Next we present the experimental studies on speech emotion recognition and speaker identification tasks using the proposed dynamic kernel (SLPMKs and SLPSKs) based ELM classifiers.

6.2 Experimental studies using dynamic kernel based ELM classifiers with the proposed SLPSKs and SLPMKs

In this section, experimental studies using proposed dynamic kernel based ELM classifiers is presented. As discussed in

Table 2 Comparison of classification accuracy (CA) (in %) using conventional ELM considering bag-of-codeword representations of speech signals as input for speech emotion recognition (SER) and speaker identification (Spk-id) tasks

K-means based codebook representation					GMM-based codebook representation					
SER			Spk-Id		SER			Spk-Id		
EmoDB	FAU-AEC				EmoDB		FAU-AEC			
(Q, h)	CA 95% CI	(Q, h)	CA 95% CI	(Q, h)	CA 95% CI	(Q, h)	CA 95% CI	(Q, h)	CA 95% CI	(Q, h)	CA 95% CI
(64, 2048)	70.82 ± 0.29	(64, 128)	**54.86 ± 0.16**	(64, 1024)	59.67 ± 0.15	(64, 2048)	75.47 ± 0.29	(64, 256)	58.86 ± 0.18	(64, 1024)	62.14 ± 0.17
(128, 2048)	69.22 ± 0.23	(128, 256)	53.60 ± 0.15	(128, 1024)	**62.19 ± 0.15**	(128, 2048)	78.72 ± 0.23	(128, 128)	**61.54 ± 0.15**	(128,1024)	**67.07 ± 0.16**
(256, 2048)	69.59 ± 0.21	(256, 256)	53.20 ± 0.17	(256, 1024)	57.13 ± 0.15	(256, 2048)	80.20 ± 0.26	(256, 256)	59.55 ± 0.19	(256, 1024)	57.17 ± 0.14
(512, 2048)	72.62 ± 0.25	(512, 512)	51.65 ± 0.13	(512, 1024)	58.02 ± 0.12	(512, 2048)	**81.37 ± 0.25**	(512, 512)	56.53 ± 0.17	(512, 1024)	62.78 ± 0.14
(1024, 2048)	**75.67 ± 0.28**	(1024, 512)	50.99 ± 0.14	(1024, 1024)	57.07 ± 0.13	(1024, 2048)	79.89 ± 0.27	(1024, 512)	56.73 ± 0.15	(1024, 1024)	62.23 ± 0.13

K-means clustering and GMM-based clustering techniques are used to obtain codebooks. Here, CA 95% CI indicates average classification accuracy along with 95% CI. Q is the number of codewords and h is the number of hidden nodes in ELM. Bold numerals indicate the best accuracy with respect to corresponding dataset and task

Sect. 2, KELM has two advantages over the conventional ELM. One of the advantage is that we need not have to consider the random weights for the input layer. Second advantage is that, we need not have to choose the nodes for the hidden layer. In this study, we use the proposed dynamic kernels in the KELM to handle the varying length patterns of speech efficiently. The classification accuracies for the ELM-based classifier using the proposed SLPSKs and SLPMKs are given in Tables 3 and 4 for speech emotion recognition and speaker identification tasks. In our studies, the dynamic kernel based ELM classifier using the SLPSK is built using different values for Q corresponding to the number of Gaussian components and N correspond to the number of segmental division. The classification accuracies for the KELM using SLPSK are given in Table 3 for speech emotion recognition and speaker identification tasks using different pooling techniques. The best performances are shown using bold phase. SLPSK computed using sum pooling performed better for speaker identification task and SLPSK computed using max pooling performed better for speech emotion recognition task.

In our studies, the ELM-based classifiers using the CBSLPMK and GMMSLPMK are built using different values for Q corresponding to the number of codewords and J corresponding to the number of levels in pyramid. In CBSLPMK, Q corresponds to number of clusters obtained using K-means clustering technique and in GMMSLPMK, Q corresponds to number of Gaussian components. The classification accuracies for the ELM-based classifier using CBSLPMK and GMMSLPMK are given in Table 4 for speech emotion recognition and speaker identification tasks. It is seen that, the ELM-based classifiers using GMMSLPMK perform significantly better than ELM-based classifiers using CBSLPMK for all the tasks. The better performance of the KELM-based classifier using the proposed dynamic kernel is mainly due to the capabilities of the SLPMKs and SLPSKs in capturing the local information better than the other dynamic kernels and also maintaining temporal information for some extent.

For all the experimental studies using dynamic kernel based ELM classifier, regularization coefficient C defined in Sect. 2, is chosen empirically as 10^{-2}. In state-of-the-art-approaches dynamic kernels are mostly used with SVM-based classifiers (Dileep and Chandra Sekhar 2012, 2014). But in our work, we have proposed dynamic kernel based ELMs for handling varying length pattern classification problem. Reason for the same is kernel ELM based classifier is comparable with SVM based classifier (Chorowski et al. 2014) and have many advantages like, it is simple, deals with multi-class classification problem and takes less training time in compare to training time of SVM based classifier. In the next Section, we present the comparison of kernel ELM with SVM using proposed dynamic kernels.

Table 3 Classification accuracy (in %) of the dynamic kernel based ELM classifiers classifier with SLPSK for speech emotion recognition (SER) and speaker identification (Spk-ID) tasks using different pooling technique for the different values of Q and N

Q	N	KELM using SLPSK with average pooling			KELM using SLPSK with sum pooling			KELM using SLPSK with max pooling		
		SER		Spk-ID	SER		Spk-ID	SER		Spk-ID
		Emo-DB	FAU-AEC		Emo-DB	FAU-AEC		Emo-DB	FAU-AEC	
		CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI
256	1	86.01 ± 0.21	65.15 ± 0.09	84.12 ± 0.18	90.80 ± 0.24	64.89 ± 0.19	82.28 ± 0.19	86.00 ± 0.18	65.08 ± 0.14	82.99 ± 0.11
	2	87.82 ± 0.23	66.17 ± 0.15	85.07 ± 0.15	90.90 ± 0.18	65.60 ± 0.17	83.03 ± 0.14	88.60 ± 0.19	65.89 ± 0.16	83.11 ± 0.16
	4	84.07 ± 0.20	65.28 ± 0.16	82.28 ± 0.19	89.60 ± 0.19	65.22 ± 0.15	84.66 ± 0.14	87.24 ± 0.18	65.01 ± 0.11	81.94 ± 0.13
512	1	88.00 ± 0.19	65.06 ± 0.14	87.56 ± 0.11	90.60 ± 0.21	64.11 ± 0.14	87.62 ± 0.17	87.68 ± 0.21	64.17 ± 0.09	86.45 ± 0.17
	2	**90.80 ± 0.29**	65.96 ± 0.11	89.89 ± 0.19	**91.40 ± 0.25**	65.08 ± 0.12	88.17 ± 0.24	87.90 ± 0.21	65.15 ± 0.11	89.01 ± 0.12
	4	89.01 ± 0.21	64.81 ± 0.12	87.44 ± 0.19	88.30 ± 0.30	65.16 ± 0.12	87.83 ± 0.13	86.37 ± 0.28	64.78 ± 0.15	87.88 ± 0.14
1024	1	88.61 ± 0.28	65.91 ± 0.09	90.22 ± 0.19	89.00 ± 0.18	65.01 ± 0.11	90.69 ± 0.12	86.06 ± 0.18	65.10 ± 0.14	88.84 ± 0.16
	2	89.90 ± 0.23	**66.12 ± 0.12**	**91.11 ± 0.12**	90.60 ± 0.17	**66.22 ± 0.12**	**91.17 ± 0.13**	**88.40 ± 0.28**	**66.06 ± 0.13**	**89.76 ± 0.12**
	4	86.22 ± 0.19	65.66 ± 0.09	88.89 ± 0.13	88.40 ± 0.17	65.22 ± 0.08	89.07 ± 0.12	85.01 ± 0.17	63.15 ± 0.09	89.11 ± 0.13

Here, CA 95% CI indicates average classification accuracy along with 95% CI. Bold numerals indicate the best accuracy with respect to corresponding dataset and task

Table 4 Classification accuracy (CA) (in %) of the ELM-based classifiers with CBSLPMK and GMMSLPMK for speech emotion recognition (SER) and speaker identification (Spk-ID) tasks for the different values of Q and J

Q	J	KELM using CBSLPMK			KELM using GMMSLPMK		
		SER		Spk-ID	SER		Spk-ID
		Emo-DB	FAU-AEC		Emo-DB	FAU-AEC	
		CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI	CA 95% CI
256	1	77.81 ± 0.25	60.11 ± 0.08	78.99 ± 0.09	86.14 ± 0.18	64.12 ± 0.09	80.11 ± 0.08
	2	79.26 ± 0.16	62.89 ± 0.11	79.11 ± 0.09	87.00 ± 0.21	65.97 ± 0.15	81.04 ± 0.14
	3	81.00 ± 0.40	63.02 ± 0.14	78.01 ± 0.07	88.09 ± 0.16	66.17 ± 0.08	78.09 ± 0.07
512	1	81.89 ± 0.19	59.85 ± 0.07	80.12 ± 0.09	87.19 ± 0.14	67.07 ± 0.06	82.10 ± 0.06
	2	**85.60 ± 0.29**	63.89 ± 0.09	81.98 ± 0.12	90.44 ± 0.16	69.13 ± 0.05	85.56 ± 0.11
	3	85.61 ± 0.23	61.87 ± 0.06	78.23 ± 0.09	**92.23 ± 0.26**	**71.12 ± 0.15**	82.78 ± 0.06
1024	1	82.48 ± 0.26	61.08 ± 0.09	81.58 ± 0.09	87.60 ± 0.18	67.01 ± 0.05	88.09 ± 0.03
	2	85.10 ± 0.21	65.39 ± 0.08	**82.96 ± 0.16**	88.77 ± 0.18	68.09 ± 0.08	**91.65 ± 0.09**
	3	84.10 ± 0.25	**66.01 ± 0.11**	82.01 ± 0.07	87.50 ± 0.22	66.89 ± 0.17	88.89 ± 0.80

Here, CA 95% CI indicates average classification accuracy along with 95% CI. Bold numerals indicate the best accuracy with respect to corresponding dataset and task

6.3 Comparison of kernel ELM with SVM using SLPMKs and SLPSKs

In this section, comparison of dynamic kernel based ELM with SVM using proposed dynamic kernels is presented. For dynamic kernel based SVM classifiers, we uses the LIBSVM (Chang and Linm 2011) tool to build the SVM classifiers. In this study, the one-against-the-rest approach is considered. The value of trade-off parameter in SVM is chosen empirically as 10^{-3}. The classification accuracies for the KELM-based classifier and SVM based classifier using the proposed SLPSKs and SLPMKs are given in Table 5 for speech emotion recognition and speaker identification tasks. The accuracies presented in Table 5 are the accuracies observed by considering SLPMKs and SLPSKs with best parameter values shown in Table 3 and Table 4. It is observed that the KELM using GMMSLPMK performs better than that of the CBSLPMK and SLPSKs for speech emotion recognition task. It is also observed that KELM using GMMSLPMK performs comparable to SLPSK with sum pooling for speaker identification task. It is seen that the SVM classifiers using SLPSK with sum pooling is marginally better than ELM using SLPSK with sum pooling for speech emotion recognition and speaker identification tasks. However, ELM classifier using GMMSLPMK perform

Table 5 Comparison of classification accuracy (CA) (in %) of the SVM-based classifiers with Kernel ELM using FK, PSK, GMMSVK, GUMIK, GMMIMK, GMMPMK, GMMSLPMK and SLPSK for speech emotion recognition (SER) task and speaker identification (Spk-ID) task

Classification model	SER					
	Emo-DB		FAU-AEC		Spk-ID	
	$Q/(J, b)/(Q, N)/(Q, J)$	CA 95% CI	$Q/(J, b)/(Q, N)/(Q, J)$	CA 95% CI	$Q/(J, b)/(Q, N)/(Q, J)$	CA 95% CI
SVM using						
FK	256	87.05 ± 0.24	512	61.54 ± 0.11	512	89.14 ± 0.15
GMMSVK	256	87.18 ± 0.29	1024	59.78 ± 0.19	512	87.93 ± 0.14
GUMIK	256	88.17 ± 0.34	1024	60.66 ± 0.10	512	90.31 ± 0.15
GMMIMK	512	85.62 ± 0.29	1024	62.48 ± 0.07	1024	88.54 ± 0.16
GMMPMK	(11,2)	88.65 ± 0.23	(5,4)	64.73 ± 0.16	(6,4)	90.26 ± 0.15
CBSLPMK	(512,2)	87.60 ± 0.20	(1024,3)	61.04 ± 0.09	(1024,2)	84.85 ± 0.16
GMMSLPMK	(512,3)	92.24 ± 0.19	(1024,2)	67.96 ± 0.10	(1024,2)	91.35 ± 0.14
SLPSK with average pooling	(512,2)	91.18 ± 0.27	(1024,2)	67.05 ± 0.18	(1024,2)	91.01 ± 0.14
SLPSK with sum pooling	(512,2)	$\mathbf{92.60 \pm 0.23}$	(256,2)	66.29 ± 0.17	(1024,1)	91.07 ± 0.16
SLPSK with max pooling	(1024,2)	91.08 ± 0.24	(1024,2)	66.78 ± 0.15	(1024,1)	90.12 ± 0.15
Kernel ELM using						
FK	256	88.23 ± 0.25	512	63.67 ± 0.10	512	88.24 ± 0.12
GMMSVK	256	89.01 ± 0.27	1024	61.23 ± 0.13	512	89.03 ± 0.14
GUMIK	256	89.11 ± 0.23	1024	61.95 ± 0.11	512	90.78 ± 0.17
GMMIMK	512	84.13 ± 0.24	1024	62.71 ± 0.09	1024	88.89 ± 0.15
GMMPMK	(11,2)	88.15 ± 0.24	(5,4)	64.13 ± 0.14	(6,4)	89.63 ± 0.18
CBSLPMK	(512,2)	85.60 ± 0.29	(1024,3)	66.01 ± 0.11	(1024,2)	82.96 ± 0.16
GMMSLPMK	(512,3)	92.23 ± 0.26	(1024,2)	$\mathbf{71.12 \pm 0.15}$	(1024,2)	$\mathbf{91.65 \pm 0.09}$
SLPSK with average pooling	(512,2)	90.80 ± 0.29	(1024,2)	66.22 ± 0.12	(1024,2)	91.17 ± 0.13
SLPSK with sum pooling	(512,2)	91.40 ± 0.25	(256,2)	65.60 ± 0.17	(1024,2)	90.88 ± 0.07
SLPSK with max pooling	(1024,2)	88.40 ± 0.28	(1024,2)	66.06 ± 0.13	(1024,2)	89.76 ± 0.12

Here, CA 95% CI indicates average classification accuracy along with 95% CI. Q indicates the number of components considered in building GMM for each class or the number of components considered in building CIGMM or the number of virtual feature vectors considered. The pair (J, b) indicates values of J and b considered in constructing the pyramid. (Q, N) indicates the number of components considered in building GMM and the number of segments in SLPSK. (Q, J) indicates the number of components considered in building GMM and the number of levels in GMMSLPMK. Bold numerals indicate the best accuracy with respect to corresponding dataset and task

better than SVM using GMMSLPMK for speech emotion recognition in FAU-AEC and in speaker identification task. Overall, it is observed that the accuracies obtained by the SVM-based classifiers and the ELM-based classifiers using dynamic kernels are close to each other

In past few years many researchers has discussed that SVM based classifiers and kernel ELM based classifiers are comparable to each other (Chorowski et al. 2014; Zhang et al. 2016). The SVM uses kernel functions to transform the data from the original input space into a highly dimensional space called the feature space, where linear separation of training samples belonging to different classes is possible. Whereas, in dynamic kernel based ELM feature mapping is known through kernel matrix and used instead of random weight for solving the problem in kernel space. Moreover, ELM has better generalization performance, better scalability and runs at much faster learning speed than traditional SVM (Huang et al. 2012). So, for further experimental studies and comparison with state-of-the-art approaches, we have considered dynamic kernel based ELM classifier only. In the next Section, we present the comparison of proposed dynamic kernel based ELM with state-of-the-art-approaches.

6.4 Comparison of proposed dynamic kernel based ELM with state-of-the-art approaches

In this section, we study comparison of proposed dynamic kernel based ELM classifier with state-of-the-art approaches. Table 6 compares the accuracies for speech emotion recognition and speaker identification tasks obtained using the GMM-based classifiers, multi-layer feed-forward neural networks (MLFFNNs), conventional

Table 6 Comparison of classification accuracy (CA) (in %) of the GMM-based classifiers, MLFFNN, conventional ELM based classifiers and KELM-based classifiers using state-of-the-art dynamic kernels for speech emotion recognition task and speaker identification task

Classification model	SER		Spk-ID
	Emo-DB	FAU-AEC	
	CA 95% CI	CA 95% CI	CA 95% CI
MLGMM	66.81 ± 0.44	60.00 ± 0.13	77.50 ± 0.12
Adapted GMM	79.48 ± 0.31	61.09 ± 0.12	83.05 ± 0.14
MLFFNN	88.12 ± 0.18	68.02 ± 0.16	89.21 ± 0.14
SVMCNN	90.70 ± 0.28	64.09 ± 0.16	–
Conventional ELM	59.25 ± 0.29	48.60 ± 0.17	59.00 ± 0.16
Conventional ELM with *K*-mean based codebook representation	75.67 ± 0.28	54.86 ± 0.16	62.19 ± 0.15
Conventional ELM with GMM-based codebook representation	81.37 ± 0.25	61.54 ± 0.15	67.07 ± 0.16
Kernel ELM using			
FK	88.23 ± 0.25	63.67 ± 0.10	88.24 ± 0.12
GMMSVK	89.01 ± 0.27	61.23 ± 0.13	89.03 ± 0.14
GUMIK	89.11 ± 0.23	61.95 ± 0.11	90.78 ± 0.17
GMMIMK	84.13 ± 0.24	62.71 ± 0.09	88.89 ± 0.15
GMMPMK	88.15 ± 0.24	64.13 ± 0.14	89.63 ± 0.18
SLPSK with average pooling	90.80 ± 0.29	66.22 ± 0.12	91.17 ± 0.13
SLPSK with sum pooling	91.40 ± 0.25	65.60 ± 0.17	90.88 ± 0.07
SLPSK with max pooling	88.40 ± 0.28	66.06 ± 0.13	89.76 ± 0.12
CBSLPMK	85.60 ± 0.29	66.01 ± 0.11	82.96 ± 0.16
GMMSLPMK	92.23 ± 0.26	71.12 ± 0.15	91.65 ± 0.09

Here, CA 95% CI indicates average classification accuracy along with 95% CI. Bold numerals indicate the best accuracy with respect to corresponding dataset and task

ELM and KELM-based classifiers using the state-of-the-art dynamic kernels mentioned in Sect. 3 and the proposed CBSLPMK, GMMSLPMK and SLPSKs with different pooling techniques.

In this study, the GMMs whose parameters are estimated using the maximum likelihood (ML) method (MLGMM) and by adapting the parameters of the UBM or class independent GMM to the data of a class (adapted GMM) (Reynolds et al. 2000) are considered to build GMM-based classifiers. The GMMs are built using the diagonal covariance matrices. MLFFNNs required fixed length input. We have converted varying length set of feature vector representation to fixed length contextual vector using the same approach explained in the Sect. 6.1 for passing the data to MLFFNNs. In our experiments for MLFFNNs architecture we have considered 3 hidden layers with 512 neurons in each layers and sigmoid activation function. We used SVM classifier with the convolutional neural network (SVMCNN) based features extracted from the architecture defined in (Mao et al. 2014). We reproduce the results for Emo-DB and FAU-AEC datasets but due to limited resources, we could not produce the results for large size speaker identification dataset. Details of conventional ELM experiments are presented in Sect. 6.1. Experiments of KELM-based classifiers are performed using the state-of-the-art dynamic kernels mentioned in Sect. 3 and the proposed CBSLPMK, GMMSLPMK and SLPSKs with different pooling techniques. The best performances are shown using bold phase. Fisher kernel (FK) using GMM-based likelihood score vectors (Smith et al. 2001), GMM supervector kernel (GMMSVK) (Campbell and Sturim 2006), GMM-UBM mean interval kernel (GUMIK) (You et al. 2009), GMM-based intermediate matching kernel (GMMIMK) (Dileep and Chandra Sekhar 2014) and GMM-based pyramid match kernel (GMMPMK) (Dileep and Chandra Sekhar 2012) are the state-of-the-art dynamic kernel based ELM classifiers considered for the study. The accuracies presented in Table 6 are the best accuracies observed among the GMM-based classifiers, MLFFNNs, conventional ELM and KELM-based classifiers with dynamic kernels using different values for their parameters. The details of the dynamic kernel based experiments and the best values for the parameters can be found in Dileep and Chandra Sekhar (2012, 2014).

7 Discussion

The ELM is a learning algorithm for single layer feed-forward networks (SLFNs) that does not involve iterative learning. An important benefit of ELM is that the hidden layer of the SLFNs need not be tuned. ELM requires less human intervention in tuning the parameter than in SVMs. In ELM, only one regularization coefficient C needs to be tuned in experiments if the feature mappings h($\mathbf{x}$) are known priorly. The feature mapping is incorporated into ELM using kernel matrix. The generalization ability of ELM is not sensitive to the dimensionality h of the ELM space (the number of hidden nodes) as long as there are sufficient number of training examples. The experimental studies show that, dynamic kernel based ELM achieve comparable or better performance to that obtained using SVM classifiers using dynamic kernels for speech emotion recognition and speaker identification tasks. ELM has better scalability and computationally efficient than traditional SVMs. The performance of the ELM-based classifiers using the proposed GMM-based SLPMK is significantly better than the SVM-based classifier and ELM-based classifier using state-of-the-art dynamic kernels for FAU-AEC dataset. For the remaining datasets, SVM classifiers using SLPSK with sum pooling is performing better

then SVM-based classifier and KELM-based classifier using state-of-the-art dynamic kernels. However, its performance is very close to that of the KELM classifier using SLPSK with sum pooling and GMM-based SLPMK. We have done experiments with MLFFNNs also for comparing with proposed approach by converting the varying length set of feature vector representation of data example to fixed length contextual vectors as per the standard defined in Chen et al. (2015). We observed that MLFFNN is performing comparable to proposed KELM using SLPMKs and SLPSKs. Tuning of parameters for MLFFNN requires huge amount of time in compare to training of KELM. For small dataset like EMO-DB and FAU-AEC building KELM based classifier is good choice in compare to MLFFNNs.

8 Conclusion and future work

In this paper, we proposed the segment-level pyramid match kernels (SLPMKs) and segment-level probabilistic sequence kernels (SLPSKs) for the classification of varying length patterns of speech represented as sets of feature vectors using ELM-based classifiers. The SLPMK is computed by partitioning the speech signal into increasingly finer subparts and matching the corresponding subparts using a segment-level pyramid. The SLPSK is computed by partitioning the speech signal into finer segments and computing the pooled probabilistic alignment vector of corresponding segment and then matching the corresponding part using a probabilistic sequence kernel. The effectiveness of the proposed SLPMKs and SLPSKs in building the KELM-based classifiers for classification of varying length patterns of long duration speech is demonstrated using studies on speech emotion recognition and speaker identification tasks.

In future work, the proposed SLPMKs and SLPSKs can also be used for classification of varying length patterns extracted from video, audio, music, and so on, represented as sets of continuous valued feature vectors using KELM-based classifiers.

References

Alexandos, I., Tefas, A., & Pitas, ioannis. (2015). On the kernel extreme learning machine classifiers. *Pattern Recognition Letters*, *54*, 11–17.

Allwein, E. L., Schapire, R. E., & Singer, Y. (2000). Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, *1*(Dec), 113–141.

Boughorbel, S., Tarel, J. P., & Boujemaa, N. (2005). The intermediate matching kernel for image local features. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2005)* (pp. 889–894), Montreal

Burkhardt, F., Paeschke, A., Rolfes, M., & Weiss, W. S. B. (2005). A database of German emotional speech. In *Proceedings of INTERSPEECH* (pp. 1517–1520), Lisbon.

Campbell, W. M., & Sturim, D. D. E. (2006). Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Processing Letters*, *13*(5), 308–311.

Chang, C. C., & Linm, C. J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1–27. http://www.csie.ntu.edu.tw/cjlin/libsvm.

Chen, Yh., Lopez-Moreno, I., Sainath, T., Visontai, M., Alvarez, R., & Parada, C. (2015). Locally connected and convolutional neural networks for small footprint speaker recognition. In *Proceedings of INTERSPEECH* (pp. 1136–1140), Dresden.

Chorowski, J., Wang, J., & Zurada, J. M. (2014). Review and performance comparison of svm-and elm-based classifiers. *Neurocomputing*, *128*, 507–516.

Dileep, A. D., & Chandra Sekhar, C. (2012). Speaker recognition using pyramid match kernel based support vector machines. *Internatiional Journal for Speech Technology*, *15*(3), 365–379.

Dileep, A. D., & Chandra Sekhar, C. (2014). GMM-based intermediate matching kernel for classification of varying length patterns of long duration speech using support vector machines. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(8), 1421–1432.

Gemert, Veenman C. J., Smeulders, A. W. M., & Geusebroek, J. M. (2010). Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(17), 1271–1283.

Gordon, G., & Tibshirani, R. (2012). Karush-kuhn-tucker conditions. *Optimization*, *10*(725/36), 725.

Grauman, K., & Darrell, T. (2007). The pyramid match kernel: Efficient learning with sets of features. *The Journal of Machine Learning Research*, *8*, 725–760.

Gupta, S., Dileep, A. D., & Thenkanidiyoor, V. (2016a). Segment-level pyramid match kernels for the classification of varying length patterns of speech using svms. In *Signal Processing Conference (EUSIPCO), 2016 24th European*, IEEE (pp. 2030–2034).

Gupta, S., Thenkanidiyoor, V., & Dileep, A. D. (2016b). Segment-level probabilistic sequence kernel based support vector machines for classification of varying length patterns of speech. In *International Conference on Neural Information Processing* (pp. 321–328). New York: Springer.

Huang, G. (2014). An insight into extreme learning machines: Random neurons, random features and kernels. *Cognitive Computation*, *6*(3), 376–390. https://doi.org/10.1007/s12559-014-9255-2.

Huang, G. B., Chen, L., & Siew, C. K. (2006). Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Networks*, *17*(4), 879–892.

Huang, G. B., Zhou, H., Ding, X., et al. (2012). Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, B (Cybernetics)*, *42*(2), 513–529.

Lee, K. A., HTK You, C. H. (2007). A GMM-based probabilistic sequence kernel for speaker verification. In *Proceedings of INTERSPEECH*, (pp. 294–297), Antwerp.

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, (vol. 2, pp. 2169–2178), New York.

Mao, Q., Dong, M., Huang, Z., & Zhan, Y. (2014). Learning salient features for speech emotion recognition using convolutional neural networks. *IEEE Transactions on Multimedia*, *16*(8), 2203–2213.

Newcombe, R. G. (1998). Two-sided confidence intervals for the single proportion: Comparison of seven methods. *Statistics in Medicine*, *17*(8), 857–872.

Rabiner, L., & Juang, B. H. (2003). Fundamentals of Speech Recognition. Pearson Education.

Rao, C. R., & Mitra, S. K. (1971). *Generalized inverse of matrices and its applications* (Vol. 7). New York: Wiley.

Reynolds, D. A. (1995). Speaker identification and verification using Gaussian mixture speaker models. *Speech Communication*, *17*, 91–108.

Reynolds, D. A., Quatieri, T. F., & Dunn, R. B. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, *10*(1–3), 19–41.

Sachdev, A., Dileep, A. D., & Thenkanidiyoor, V. (2015). Example-specific density based matching kernel for classification of varying length patterns of speech using support vector machines. In *Proceedings of ICONIP*, (pp. 177–184). Istanbul.

Smith, N., Gales, M., & Niranjan, M. (2001). Data-dependent kernels in SVM classification of speech patterns. Tech. Rep. CUED/F-INFENG/TR.387, Cambridge University Engineering Department, Cambridge.

Steidl, S. (2009). Automatic classification of emotion-related user states in spontaneous childern's speech. PhD thesis, Der Technischen Fakultät der Universität Erlangen-Nürnberg, Germany.

Swain, M. J., & Ballard, D. H. (1991). Color indexing. *International Journal of Computer Vision*, *7*(1), 11–32.

The NIST Year 2002 Speaker Recognition Evaluation Plan. (2002). http://www.itlnistgov/iad/mig/tests/spk/2002/

The NIST Year 2003 Speaker Recognition Evaluation Plan. (2003). http://www.itlnistgov/iad/mig/tests/sre/2003/

Vedaldi, A., & Zisserman, A. (2010). Efficient additive kernels via explicit feature maps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 3539–3546).

Wang J., KYFLTH Yang, J., & Gong, Y. (2010). Locality-constrained linear coding for image classification. In *Proceedings of CVPR'10*, IEEE (pp. 3360–3367). State College: The Pennsylvania State University.

Yang, J., Yu, K., Gong, Y., & Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of CVPR'09*, IEEE, (pp. 1794–1801).

You, C. H., Lee, K. A., & Li, H. (2009). An SVM kernel with GMM-supervector based on the Bhattacharyya distance for speaker recognition. *IEEE Signal Processing Letters*, *16*(1), 49–52.

Zhang, L., Zhang, D., & Tian, F. (2016). Svm and elm: Who wins? object recognition with deep convolutional features from imagenet. In *Proceedings of ELM-2015* (Vol. 1, pp. 249–263). Springer: New York.